HACK X CRACK: INSEGURIDAD POR DCC // NO TE FIES !!!

PC PASO a PASO

Enfréntate al futuro

XML



EL ESTANDAR UNIVERSAL

Visual Basic
Acceso a Datos

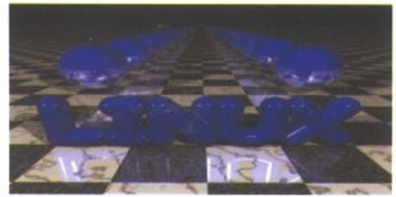


Organízate!!!

TRIO DE ASES:
APACHE + PHP + MYSQL

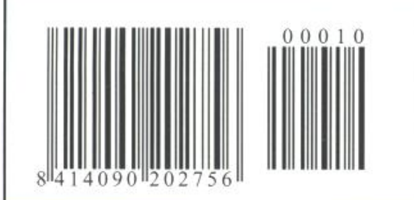


PHPBB: MONTA TU PROPIO FORO



Gestión de Usuarios y el estándar VI

Nº 10 -- P.V.P. 4,5 EUROS





PROTOCOLO DCC
DIRECT CLIENT TO CLIENT

Escaneo mediante Fserve

Obtención de IPs

INSEGURO Y PELIGROSO

HACKEANOS !!!

IP Spoofing en DCC

DCC Send Hijacking

"Puenteando" un DCC

Obtención de Puertos

SOLO EXISTE UNA DEFENSA ANTE LOS ATAQUES EN INTERNET:
LEER ESTA PUBLICACÓN CADA MES... DEJA DE SER UNA VÍCTIMA !!!

PC PASO A PASO: PREPÁRATE PARA EL FUTURO CON NOSOTROS

HACK X CRACK - HACK X CRACK - HACK X CRACK

EDITORIAL

INTENTANDO MEJORAR

Estimados lectores, los colaboradores de Hack x Crack estamos trabajando duro para ofreceros cada mes nuestros conocimientos e intentamos no caer en la tentación de convertirnos en una revista del tipo "aprieta este botón y conviértete en un hacker".

Lo que intentamos cada mes es alternar la "carnaza práctica" con el "verdaderos conocimiento". Sería muy fácil explicar cada mes un par de exploits, alentarte a explotar vulnerabilidades y dejarte caer en la ignorancia, muy al contrario, hemos elegido el camino difícil: "obligarte" a estudiar un poco en cada número.

Para después del verano estamos preparando una ampliación de las páginas, inclusión de publicidad, instalación de más servidores y muchas cosas más. Ya sabes que somos lentos puesto que no disponemos de recursos económicos, pero también sabes que poco a poco cumplimos nuestras promesas. Muy poco a poco, pero no podemos hacer mucho más de lo que hacemos.

Espero que te guste el Curso de XML que iniciamos con este número 10 y te "advertimos" que el futuro de la Informática pasa necesariamente por XML.

Dentro de poco MICROSOFT te dirá que ha inventado XML y que Office 2003 es claro ejemplo de su inventiva. No, no te dejes engañar, XML hace bastante que fue "inventado" y desde luego no fue Microsoft quien lo hizo ;)

Solo puedo acabar de una manera, agradeciendo a cuantos nos leen su fidelidad y agradeciendo a cuantos colaboran con nosotros por su entrega incondicional.

Una vez más GRACIAS!

INDICE

SERVIDOR DE HXC MODO DE EMPLEO

- Hack x Crack ha habilitado un servidor para que puedas realizar las prácticas de hacking.

- Actualmente tiene el BUG del Code / Decode y lo dejaremos así por un tiempo (bastante tiempo ;) Nuestra intención es ir habilitando servidores a medida que os enseñemos distintos tipos de Hack, pero por el momento con un Servidor tendremos que ir tirando (la economía no da para mas).

- En el Servidor corre un Windows 2000 Advanced Server con el IIS de Servidor Web y está en la IP 80.36.230.235.

- El Servidor tiene tres unidades:
* La unidad c: --> Con 2GB
* La unidad d: --> Con 35GB y Raíz del Sistema
* La unidad e: --> CD-ROM

Nota: Raíz del Servidor, significa que el Windows Advanced Server está instalado en esa unidad (la unidad d:) y concretamente en el directorio por defecto \winnt\ Por lo tanto, la raíz del sistema está en d:\winnt\

- El IIS, Internet Information Server, es el Servidor de páginas Web y tiene su raíz en d:\inetpub (el directorio por defecto)

Nota: Para quien nunca ha tenido instalado el IIS, le será extraño tanto el nombre de esta carpeta (d:\inetpub) cómo su contenido. Pero bueno, un día de estos os enseñaremos a instalar vuestro propio Servidor Web y detallaremos su funcionamiento.

De momento, lo único que hay que saber es que cuando TÚ pongas nuestra IP (la IP de nuestro servidor) en tu navegador, lo que estás haciendo realmente es ir al directorio d:\Inetpub\wwwroot\ y leer un archivo llamado default.htm.

Nota: Como curiosidad, te diremos que APACHE es otro Servidor de páginas Web (seguro que has oído hablar de él). Si tuviésemos instalado el apache, cuando pusieses nuestra IP en TU navegador, accederías a un directorio raíz del Apache (donde se hubiese instalado) e intentarías leer una página llamada index.html

Explicamos esto porque la mayoría, seguro que piensa en un Servidor Web como en algo extraño que no saben ni donde está ni como se accede. Bueno, pues ya sabes dónde se encuentran la mayoría de IIS (en \Inetpub\) y cuál es la página por defecto (\Inetpub\wwwroot\default.htm). Y ahora, piensa un poco... ... ¿Cuál es uno de los objetivos de un hacker que quiere decirle al mundo que ha hackeado una Web? Pues está claro, el objetivo es cambiar (o sustituir) el archivo default.html por uno propio donde diga "hola, soy DIOS y he hackeado esta Web" (eso si es un lamer ;)

A partir de ese momento, cualquiera que acceda a ese servidor, verá el default.htm modificado para vergüenza del "site" hacheado. Esto es muy genérico pero os dará una idea de cómo funciona esto de hackear Webs ;)

- Cuando accedas a nuestro servidor mediante el CODE / DECODE BUG, crea un directorio con tu nombre (el que mas te guste, no nos des tu DNI) en la unidad d: a ser posible (que tiene mas espacio libre) y a partir de ahora utiliza ese directorio para hacer tus prácticas. Ya sabes, subirnos programitas y practicar con ellos :)

Puedes crearte tu directorio donde quieras, no es necesario que sea en d:\mellamojuan. Tienes total libertad!!! Una idea es crearlo, por ejemplo, en d:\winnt\system32\default\mellamojuan (ya irás aprendiendo que cuanto mas oculto mejor :)

Es posiblemente la primera vez que tienes la oportunidad de investigar en un servidor como este sin cometer un delito (nosotros te dejamos y por lo tanto nadie te perseguirá). Aprovecha la oportunidad!!! e investiga mientras dure esta iniciativa (que esperamos dure largos años)

- En este momento tenemos mas de 600 carpetas de peña que, como tu, está practicando. Así que haznos caso y crea tu propia carpeta donde trabajar.



MUY IMPORTANTE...

MUY IMPORTANTE!!!!! Por favor, no borres archivos del Servidor si no sabes exactamente lo que estás haciendo ni borres las carpetas de los demás usuarios. Si haces eso, lo único que consigues es que tengamos que reparar el sistema servidor y, mientras tanto, ni tu ni nadie puede disfrutar de él :(Es una tontería intentar "romper" el Servidor, lo hemos puesto para que disfrute todo el mundo sin correr riesgos, para que todo el mundo pueda crearse su carpeta y practicar nuestros ejercicios. En el Servidor no hay ni Warez, ni Programas, ni claves, ni nada de nada que "robar", es un servidor limpio para TI, por lo tanto cuídalo un poquito y montaremos muchos más :)

GNU LINUX (III) GESTION DE USUARIOS EDITORES DE TEXTO: "VI"

Gestión de Usuarios en Linux: Crearemos Usuarios Nuevos, les pondremos Claves de Acceso y daremos un Directorio de Trabajo. De paso, examinaremos los famosos /etc/passwd y /etc/shadow. IMPRESCINDIBLE !!!

Editor VI: Mucho más que un Editor y un estándar en todas las distribuciones LINUX.

0.- Introducción

Tras una presentación general del S.O. que vamos a manejar y de mostrar las particularidades de este sistema a la hora de tratar con la información que almacena, hoy continuaremos hacia adelante con el objetivo de llegar a disponer de nuestro GNU/LiNUX como una plataforma de desarrollo.

Los puntos que hoy abarcaremos, y sin perder nunca de vista lo visto anteriormente serán: Crear y administrar usuarios y el editor *vi* . Tras esto, en el próximo artículo trataremos la programación en ***bash-shell*** y en ***C*** bajo GNU/LiNUX. Ambos temas formaban parte de este artículo, pero la extensión ocupaba cerca de las 35 páginas en A4.

1. Administración de usuarios

La tarea más básica de administración que debemos de realizar en un entorno multiusuario es siempre la administración de los posibles usuarios de dicho sistema. Entenderemos por administración de usuarios a la capacidad de crear, modificar y eliminar cuentas de ***usuario***.

Una cuenta de usuario será un conjunto de informaciones (nombre o ***login*** del usuario y clave o ***password***) relativas a una persona que le permitirán acceder a nuestro sistema.

El usuario encargado de la administración del resto de los usuarios, será como siempre ***root*** (administrador).

1.1. ¿Cómo se almacena esta información en el sistema?

Como hemos dicho, cada persona debe poseer su nombre y clave para acceder a nuestro sistema. Este nombre, a partir de ahora ***login***, debe de ser único. Es decir, distintos usuarios no pueden tener el mismo ***login***.

Toda la información relativa a las cuentas de usuario reside en el archivo ***/etc/passwd***. Este archivo debe de tener permiso de escritura para ***root*** y de lectura para el resto:

```
-rw-r--r--    1 root root  2,0K 2003-04-29 02:03
/etc/passwd
```

Si miramos el contenido de este archivo veremos algo como:

```
root@el_chaman:~# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
```

```
bin:x:2:2:bin:/bin:/bin/sh
tuxed:x:1001:1001:,,,:/home/tuxed:/bin/bash
perex:x:1013:1013:,,,:/home/perex:/bin/bash
bea:x:1010:1010:,,,:/home/bea:/bin/bash
academia:x:1022:1022:,,,:/home/academia:/bin/bash
luis:x:1004:1004:,,,:/home/luis:/bin/bash
```

Cada una de las líneas posee un formato predefinido y se interpretan de la siguiente manera:

```
usuario:password:ID_usuario:ID_grupo:comentario:directorio_home:comando
```

Donde cada uno de los campos significan:

usuario: El login del usuario. Debe de ser distinto para cada usuario.

password: La clave para el usuario encriptada.

ID_usuario: Un número que identifica al usuario frente al S.O. Cada número es exclusivo de un usuario.

ID_grupo: Un número que identifica al grupo al que pertenece el usuario ante el S.O. Cada número es exclusivo de un grupo.
comentario: Un comentario sobre el usuario: puede ser el puesto, nombre real, etc...

directorio_home: El directorio home del usuario.

comando: Comando a ejecutar cuando el usuario accede al sistema. Normalmente es una *shell*.

Llegados a este punto, debemos de hacer algún comentario sobre los nombres de usuario y las claves.

Normalmente se recomienda que los nombres de usuarios sean cadenas de caracteres de hasta ocho caracteres como máximo. A pesar de que las mayúsculas, los subrayados u otros caracteres especiales están permitidos, no se recomienda su uso. Esto es más debido a una costumbre nacida de la costumbre práctica de los sistemas UNiX que una regla obligatoria.

Con respecto a las claves voy a ser un poco más extenso dado que hay mucho más que contar.

Tal vez a alguien le haya llamado la atención encontrarse una x en el campo password en el fichero arriba mostrado. Incluso puede ser que alguien haya llegado a la conclusión de que el autor ha puesto una x en lugar de la clave encriptada "por si las moscas". Pues bien, esa no es la razón. En breve veremos el porqué de esa x.

Como se ha dicho ya un par de veces, el sistema guarda la clave encriptada en el campo ***password*** en un sistema GNU/LiNUX estándar. este campo puede ser modificado mediante la invocación del comando ***passwd***, ya sea por el administrador para cambiar la clave de cualquier usuario, ya sea por cualquier usuario para cambiar su propia clave.

Dicho esto, nos encontramos con una peculiaridad que podría ser fuente de muchos problemas: el archivo ***/etc/passwd*** tiene permiso de lectura **para todos** los usuarios del sistema. Adivinad cual es el alimento de los campeones para Juanito el Ripeador.



Para quien no sepa...

Para quien no sepa qué es eso de "Juanito", necesita pasarse urgentemente por http://www.openwall.com/john/ y http://www.decowar.com/manual_john_the_ripper.htm Un día de estos nos meteremos de lleno en este tema, pero de momento, en Internet hay mucha información al respecto :)

Algunos sistemas UNIX, intentando evitar este

problema, almacenan las claves encriptadas en otro archivo, ***/etc/shadow***, el cual sólo es accesible para el administrador y el grupo ***shadow***. Se indica la existencia de un archivo ***/etc/shadow*** si en ***/etc/passwd*** encontramos una x en el campo ***password***.

Ojo: Insisto una vez más porque es muy importante; sólo se puede cambiar la clave mediante el comando ***passwd***: No intentéis editar el archivo /etc/passwd o /etc/shadow "a mano".

1.2. Creando un usuario

Antes de nada, como vamos a trastear con cosas un poco peligrosas, conviene cubrirnos las espaldas y generar unas copias de seguridad:

```
root@el_chaman:~# mkdir bck
root@el_chaman:~# cp /etc/passwd* bck
root@el_chaman:~# cp /etc/shadow* bck
root@el_chaman:~# cp /etc/group* bck
root@el_chaman:~# ls bck
total 36K
4,0K group  4,0K group.org  4,0K passwd-  4,0K shadow  4,0K shadow.org
4,0K group-  4,0K passwd  4,0K passwd.org  4,0K shadow-
root@el_chaman:~#
```

Si algo fuera realmente mal, bastaría con teclear:

```
root@el_chaman:~# cp bck/* /etc
```

Para dejar las cosas como estaban.

A la hora de crear un nuevo usuario dispondremos de dos maneras: Cambiando manualmente todo lo necesario para crearlo o utilizando diversos ***scripts*** o utilidades que vienen en las distintas distribuciones (por ejemplo ***adduser***, ***script*** en Perl bastante común, o ***kuser***, programa gráfico de KDE)

Ni que decir tiene que nosotros optaremos por hacerlo a mano en este artículo por una simple razón: Queremos saber qué pasa en nuestro sistema cuando se añade un usuario.

Los pasos generales que seguiremos para crear un archivo serán:

1 - Añadir la información adecuada a nuestro archivo ***/etc/passwd***.

2 - Crear un directorio home para el usuario y asignarle como propietario el nuevo usuario.

3 - Copiar los archivos de configuración necesarios a dicho directorio y asignarlos los permisos/propietarios adecuados.

Añadiendo información

Veamos un ejemplo. Vamos a añadir el usuario hxc. A este usuario le daremos como clave 123hXc321 y le asignaremos como directorio home ***/home/hxc***

Siguiendo el guión mostrado, la primera tarea a realizar será la de añadir como root una entrada a ***/etc/passwd***. Esto lo podremos hacer con un editor como el ***vi*** que veremos más adelante en este mismo artículo. Nosotros añadiremos:

```
hxc::1024:100::/home/hxc:/bin/bash
```

hxc será el nombre del usuario. ***El campo de password queda vacío***. usuario_ID será ***1024*** o cualquier otro por encima de ***1000*** que no esté siendo utilizado. grupo_ID será users (***100***). El directorio home estará en ***/home/hxc*** y finalmente al entrar se ejecutará un ***bash shell***.

A continuación añadiremos el password. En un sistema UNiX normal bastaría con:

```
root@el_chaman:~# passwd hxc
Enter new UNIX password:
```

```
Retype new UNIX password:
passwd: password updated successfully
```

Pero si queremos utilizar shadow-passwords será:

```
root@el_chaman:~# shadowconfig on
Shadow passwords are now on.
root@el_chaman:~# passwd hxc
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

Recomiendo examinar las entradas de ***/etc/passwd*** y de ***/etc/shadow*** en cada caso para ver los cambios efectuados

Creando el directorio home

Ahora, creamos el directorio home para el usuario hxc:

```
root@el_chaman:~# mkdir /home/hxc
```

Una vez hecho esto, cambiamos el propietario de dicho directorio:

```
root@el_chaman:~# chown hxc.users /home/luis
```

Copiando archivos de configuración necesarios

Estos archivos normalmente suelen ser ***/etc/bashrc, /etc/profile*** y todo aquello que queramos meter. Un ejemplo clásico será:

```
root@el_chaman:~# cp /etc/bashrc /home/hxc/.bashrc
root@el_chaman:~# cp /usr/share/vim/vimrc /home/hxc/.vimrc
root@el_chaman:~# chown hxc.users /home/hxc/.bashrc
root@el_chaman:~# chown hxc.users /home/hxc/.vimrc
```

y editaríamos dichos archivos más tarde a nuestro gusto. Obsérvese que ambos archivos los dejamos ocultos (empiezan por un punto) y cambiamos su propietario a hxc.

Como es obvio que esto resulta tedioso, normalmente lo que se hace es crear un directorio /etc/skel donde depositamos todos los archivos de configuración o iniciales de los que dispongan los nuevos usuarios. Posteriormente se copian estos archivos al directorio home, se actualizan los permisos y listo.

Borrando un usuario

Borrar un usuario consistirá en desandar el camino recorrido para crearlo: Básicamente tendremos que eliminar su directorio home y borrar las entradas necesarias en ***/etc/passwd y /etc/shadow*** si corresponde.

Insisto en que todas las tareas vistas actualmente las realizan diversos scripts y programas que nos facilitan mucho la vida. A veces estos programas varían de una distribución a otra, pero ***ABSOLUTAMENTE TODOS ELLOS*** realizan las tareas que hemos descrito.

1.3. Grupos

Los grupos funcionan de manera análoga a los usuarios. Cada usuario debe de pertenecer a un grupo y un grupo puede gestionar los permisos de varios usuarios simultaneamente (recordad en el artículo anterior cómo asignábamos permisos al propietario, al grupo y al resto de los usuarios). Resultan una manera muy cómoda de manejar sistemas en los que existen muchos usuarios que realizan tareas muy diferentes.

La información relativa a los grupos se encuentra en ***/etc/group*** y suele tener el siguiente aspecto:

```
......
proxy:x:13:
kmem:x:15:
dialout:x:20:
```

```
fax:x:21:
voice:x:22:
cdrom:x:24:
floppy:x:25:
tape:x:26:
sudo:x:27:
audio:x:29:
dip:x:30:
postgres:x:32:
www-data:x:33:
backup:x:34:
operator:x:37:
list:x:38:
irc:x:39:
src:x:40:
gnats:x:41:
shadow:x:42:
utmp:x:43:telnetd
video:x:44:
staff:x:50:
users:x:100:luis, hxc
.

.....
```

Cada línea se interpreta de la siguiente forma:

nombre_grupo:password_grupo:ID_grupo:usuarios_pertenecientes_a_ese_grupo

nombre_grupo: El nombre que queramos dar al grupo.

password_grupo: La clave para el grupo. Normalmente se deja en blanco o se pone un asterisco. No todos los UNiX soportan la asignación de claves a grupos y se mantiene este formato por compatibilidad con versiones más antiguas.

ID_grupo: Número que identificará al grupo ante el S.O.

usuarios_pertenecientes_a_ese_grupo: lista de todas los usuarios que pertenecen a ese grupo.

Crear un nuevo grupo y añadirle usuarios será algo tan sencillo como añadir las entradas necesarias a /etc/group.

Y con esto queda contemplado el sistema de administración de altas y bajas de usuarios. A continuación, tras ver una de las peculiaridades de los sistemas GNU/LiNUX vamos a cambiar un poco de tercio y vamos a examinar una de las herramientas más importantes en cualquier sistema UNiX: el editor ***vi***.

2. El editor vi

Muchos pensarán que escribir un artículo o parte de este explicando el manejo del vi es una pérdida de tiempo y de espacio. Personalmente voy a defender de forma breve el porqué de aprender a manejar este editor de texto.

Se supone que los lectores de esta revista son ante todo gente curiosa que le gusta investigar diferentes sistemas. Si nos centramos en los sistemas UNiX-like, nos encontramos que distintos SS.OO UNiX-like poseen diferentes herramientas o diferentes maneras de hacer las cosas. Esto muchas veces se convierte en una situación tediosa de "vuelta a empezar de 0".

Por eso se agradece que herramientas, que en principio puedan parecer poco interesantes como los editores de texto, estén disponibles en todos los entornos y se manejen de la misma manera. Éste es el caso del vi. Imaginaos que cada vez que entrásemos en una RedHat, en una Debian, en Solaris, en SCO UNiX, en FreeBSD, etc... tuviésemos que aprender cómo se maneja el editor de texto; hasta pasados unos treinta minutos no nos familiarizaríamos con él. Y treinta minutos, a pesar de ser mucho tiempo en según que situaciones, no dan mucho de sí.

Dicho esto alguno estará pensando "yo tengo el vim, ¿es lo mismo que el vi?". Sí y no. Vim (***Vi improbed***, vi mejorado) es una versión mejorada del vi que proporciona muchas mejoras al primitivo vi y es compatible con él. Todo lo que aquí se diga es válido para el vim, pero tened en cuenta que lo que vais a encontrar en todos los sistemas UNiX es el vi.

El vi (***Visual Interface***) es una mejora más que se hizo sobre los primitivos editores de líneas (como el ***ex*** o ***ed***) presentes en UNiX, y que dotaron de capacidades visuales a estos primitivos editores de manera que pudieran aprovechar las potentes capacidades de los entonces modernos terminales ***tty***.

El vi pues, no será el clásico editor de texto plano como el EDIT del MS-DOS, sino que tendrá muchas peculiaridades, la mayoría heredades del editor de líneas que siempre corre bajo el vi, los anteriormente mencionado ex y ed.
vi posee tres modos de trabajo: modo comando, modo edición y modo ex.

En el modo comando podremos movernos por el texto, invocar comandos del vi, invocar a ed o ex, etc...

En el modo edición simplemente nos limitaremos a introducir texto como si de un editor normal se tratase.

En el modo ex o ed pasamos a manejar los antiguos editores de línea.

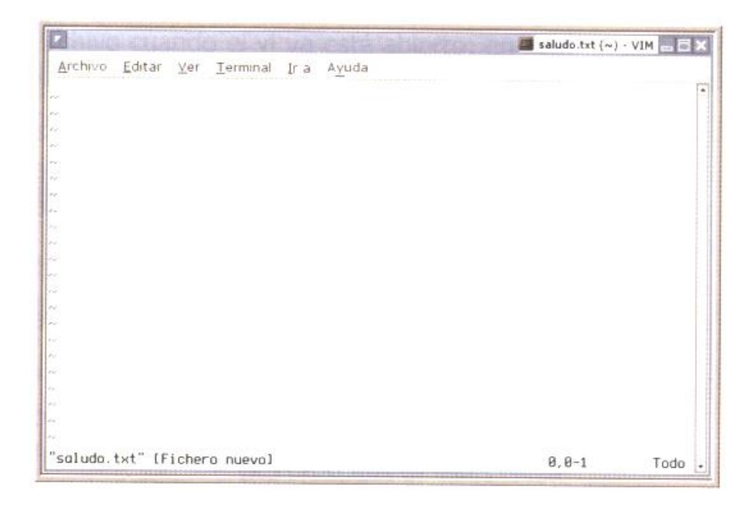
Normalmente una sesión de vi comienza invocando el propio programa seguido de uno o más ficheros. Las siguientes invocaciones de vi son válidas:

luis@el_chaman:~$ vi
luis@el_chaman:~$ vi hola.txt
luis@el_chaman:~$ vi hola.txt adios.txt

Una vez abierto el vi estaremos en modo comando. Para alternar entre modo comando y modo edición se utilizará la tecla escape. Vamos a crear un archivo llamado saludo.txt. Inicialmente invocaremos al editor de la siguiente manera:

luis@el_chaman:~$ vi saludo.txt

Se nos mostrará entonces la siguiente pantalla

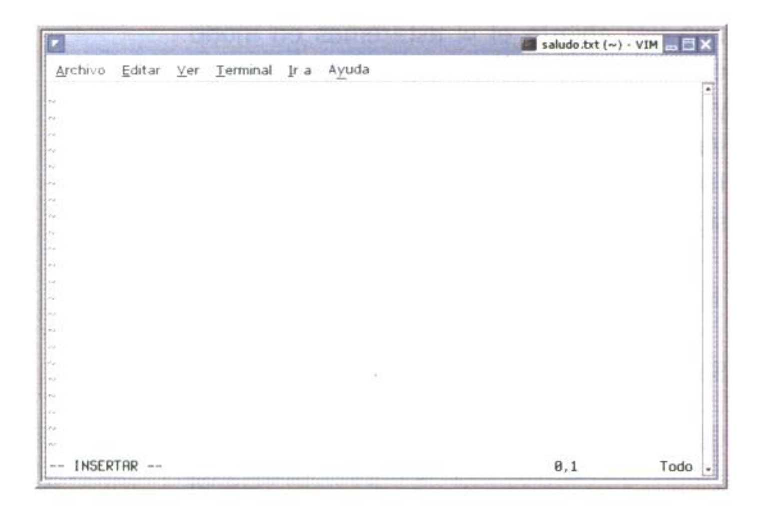


Vemos que en la parte inferior se nos muestra cierta información sobre el archivo:

"saludo.txt" [Fichero nuevo] 0,0-1 Todo

La parte izquierda es de significado obvio; los números 0-0 indican la línea y columna donde está situado el cursor, y el Todo nos dice que está mostrando todo el documento en la pantalla. Si el documento fuese más grande, en vez de poner Todo pondría el porcentaje correspondiente al texto mostrado en pantalla con respecto al total.

En este momento estamos en modo comando y el vi está preparado para que le demos una orden. La orden que vamos a dar es la de insertar texto así que pulsamos la letra '*i*' (i minúscula). Entonces obtendremos los que muestra la siguiente imagen.



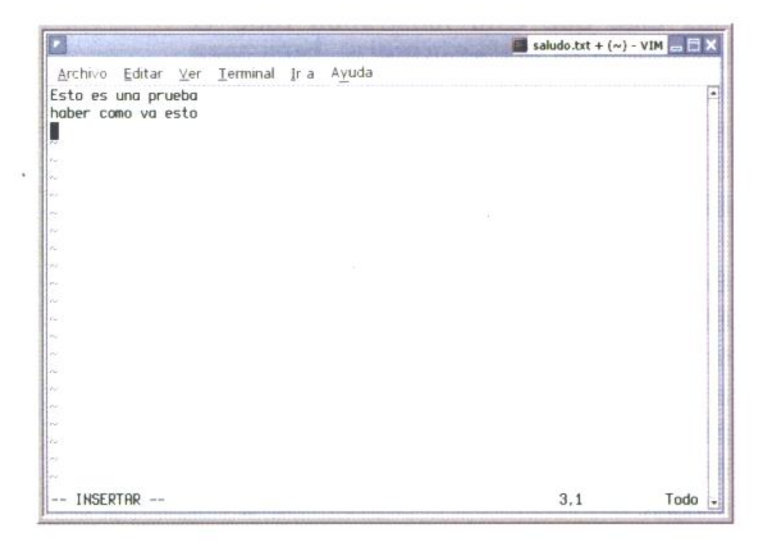
Observamos que en la parte inferior izquierda aparece la palabra -- INSERTAR --. Esto quiere decir que el vi ha entendido la orde y que estamos en modo edición.

A continuación vamos a meter el siguiente texto:

Esto es una prueba [ENTER]
haber como va esto.[ENTER]

Antes de continuar; el "haber" de "haber como va esto", es una falta de ortografía puesta con la intención de corregirla más adelante.

Una vez hecho lo indicado habremos obtenido algo como lo mostrado en la siguiente imagen.



Ahora tenemos que guardar el archivo. Para ello tenemos que pasar primero a modo comando e invocar el comando de escritura. Para pasar a modo comando damos a la tecla ESCAPE, al hacerlo vemos como el -- INSERTAR -- de la parte inferior izquierda desaparece. Ahora teclemamos :w (dos puntos uve doble) tal y como muestra la siguiente imagen.



Si presionamos ENTER nos saldrá un mensaje en la parte inferior similar a este:

"saludo.txt" [Nuevo] 3L, 39C escritos 3,0-1 Todo

Nos informa de que los cambios han sido guardados con éxito.

Para salir ahora, volvemos a presionar ESCAPE (volvemos a pasar a modo comando), y tecleamos ***:q*** (dos puntos q)

Para verificar que hemos escrito el archivo, tecleemos:

```
luis@el_chaman:~$ cat saludo.txt
Esto es una prueba
haber como va esto

luis@el_chaman:~$
```

Bien, hemos visto que este editor consta básicamente de, por un lado, enviar órdenes al propio vi (modo comando) , y por otro editar

texto (modo edición).

Vamos a ir afinando un poco más. Nuestro objetivo ahora será corregir la falta que hemos cometido. Para ello abriremos el fichero, nos situaremos sobre la palabra que queremos modificar, la borraremos y escribiremos la nueva palabra.

Empezamos por invocar al vi:

```
luis@el_chaman:~$ vi saludo.txt
```

Recordemos que inicialmente estamos siempre en modo comando. Ahora nos situamos sobre la "h" de "haber". Muchos lo podréis hacer sin problema con las flechas de los teclados modernos. Esto no valdrá para todos los sistemas, así que más vale que vayamos practicando con la siguiente sustitución:

h: mover un carácter a la izquierda
j: mover una línea abajo
k: mover una línea arriba
l: mover un carácter a la derecha

Otra sustitución útil en los casos de los que no dispongamos de las teclas Av Pág y Re Pág. será:

Ctrl + U: Retroceder media pantalla
Ctrl + D: Avanzar media página
Ctrl + F: Avanzar una pantalla
Ctrl + B: Retroceder una pantalla

Una vez situados sobre la "h" de "haber" presionaremos la "x". Observamos que sucesivas presiones en la letra x eliminan una letra. Eliminad "haber".

Ahora, queremos insertar la corrección, tecleamos entonces la letra "i" (insertar) y automáticamente pasamos al modo edición; ahora el vi se comportará como un editor de textos normal. Modificamos nuestro texto de manera que quede:

```
Esto es una prueba
a ver como va esto
```

Y volvemos a guardar el archivo invocando primeramente el modo comando (ESCAPE) y tecleando el comando :wq (escribir y salir).

Si quisiéramos guardar el archivo con otro nombre, podríamos meter el comando :w nombre_archivo.
Esto es también válido también para cargar un archivo cuando el vi ya está abierto; en este caso el comando correspondiente será :r nombre_archivo

Recapitulando:

* El vi, a la hora de editar tiene dos modos: modo comando y modo edición.

* En el modo comando nos podemos mover por el texto tanto con las flechas o las teclas Av Pág y Re Pág o con sus sustitutas: h, j, k, l, Ctrl+U, Ctrl+D, Ctrl+F y Ctrl+D.

* Alternaremos entre ambos modos pulsando la tecla ESCAPE

* Los comandos básicos a la hora de editar serán: i para insertar, x para borrar.

* Los comandos básicos a la hora de cargar/guardar archivos son respectivamente :r/:w; ambos admiten como parámetro un nombre de archivo.

Con el sencillo ejemplo visto ya seremos capaces de editar cualquier fichero de texto con el vi. Pero el vi tiene muchas más posibilidades.

Me he permitido incluir una referencia rápida de los comando del vi y un manual bastante completo. Ambos se pueden descargar respectivamente de:

http://users.servicios.retecal.es/luis-ubaldo/vi-ref.pdf

y

http://users.servicios.retecal.es/luis-ubaldo/ManualVi.pdf

Espero que con la ayuda de estos documentos y un poco de tiempo, no tardéis mucho en manejar con soltura este editor.

Y ya para terminar decir que estamos listos para afrontar la programación. Empezaremos en el próximo artículo con la programación en bash-shell y C. Advierto que la programación en C no se centrará en el lenguaje C, si no en el manejo de las distintas herramientas de las que dispondremos en GNU/LiNUX para realizar programas en C/C++ (IDEs, compiladores, depuradores, etc....).

SERIE RAW: CONOCIENDO PROTOCOLOS Y SU SEGURIDAD

RAW 4: DCC DIRECT CLIENT TO CLIENT PROTOCOL

Funcionamiento del Protocolo DCC
Codificación de Decodificación de IPs
DCC --> Inseguro y Peligroso:

- -- Obtención de IPs
- -- Obtención de Puertos
- -- DCC Send Hyjacking
- -- "FXP" en DCC
- -- Puenteando un DCC
- -- Escaneo mediante Fserver

0.- Introducción

Como prometí en el número anterior, aquí está el artículo sobre DCC, para completar todo lo referente a los protocolos relacionados con IRC.

Este artículo va a ser bastante diferente a los anteriores. Para empezar, no se basa en ningún RFC ya que, tal y como podemos comprobar buscando la palabra "DCC" en http://www.rfc-editor.org/rfcsearch.html , no existe ningún RFC que especifique el funcionamiento de este protocolo. El funcionamiento del protocolo DCC es muy simple, por lo que para detallarlo sólo emplearé la primera mitad del artículo. Es en la segunda mitad del artículo donde se encuentra la principal diferencia con otros artículos de la serie, puesto que esta parte estará enteramente dedicada a técnicas para explotar este protocolo.

Por tanto, en esta ocasión no me limitaré tan sólo a detallar el funcionamiento de un protocolo, si no también a detallar sus problemas de seguridad.

Todas estas "técnicas" para abusar del DCCfueron ideadas por mí, aunque seguro que a más de un lector, en cuanto lea la primera parte en la que explico el funcionamiento básico del protocolo, se le ocurrirán las mismas ideas, ya que son bastante simples (aunque no por ello menos efectivas). Por tanto, todo lo aquí contado es el fruto de mis experimentos personales, así que puedo garantizar que todo lo explicado funciona y, además, que ningún animal fue dañado para realizar los experimentos en el "laboratorio". ;-)

1.- FUNCIONAMIENTO DEL DCC

1.1. Funcionamiento básico del DCC a grandes rasgos

Para los que aún no sepan de qué demonios trata este artículo, os explicaré rápidamente en qué consiste el DCC.
Los usuarios de **IRC** (**I**nternet **R**elay **C**hat) no sólo tienen la posibilidad de conversar con otros

usuarios de la misma red, si no que además pueden hacer muchas otras cosas, como enviar archivos, montar servidores de archivos, chat por voz, utilizar una pizarra común para escribir o dibujar, o incluso videoconferencia. Todas estas virguerías se consiguen mediante un protocolo encapsulado en el propio protocolo de IRC, que es el DCC.

En realidad, la mayoría de estas virguerías son sólo implementadas por unas pocas aplicaciones de IRC, y están muy poco extendidas. Como sabrá cualquier asiduo al IRC, las dos utilidades típicas del DCC son: DCC Chat y DCC Send.

Ambos tienen en común una característica, y es que funcionan mediante una conexión punto a punto entre los dos usuarios, y no a través del servidor de IRC. Por eso precisamente el protocolo se llama **D**irect **C**lient to **C**lient. ;-)

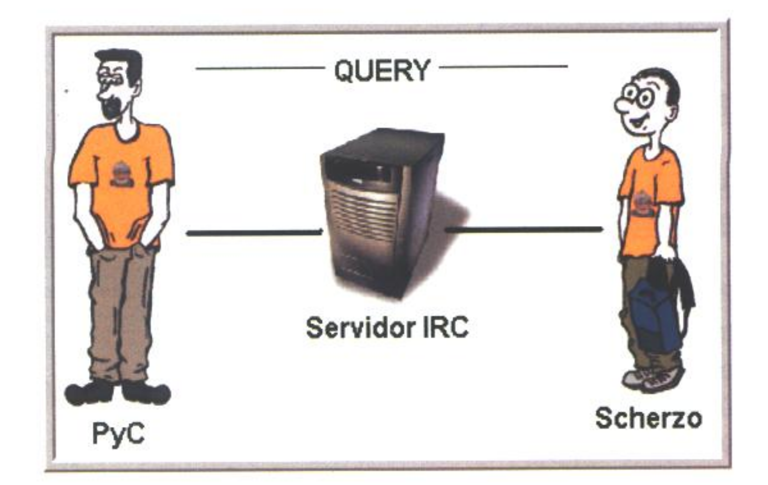
Os recuerdo que en IRC todo circula a través del servidor. Por ejemplo, cuando escribimos un mensaje privado a un usuario, en realidad estamos enviando este mensaje primero al servidor de IRC, para que luego el servidor lo envíe a su vez al usuario correspondiente.

Pensad en el problema que sería si esto no fuese así. Para empezar, en algunas redes (como en el IRC-Hispano) las IPs de los usuarios no son visibles, por lo que sería imposible establecer ningún tipo de conexión entre 2 usuarios sin la intervención del servidor, que es el único que conoce las IPs de todos los usuarios. Como segundo ejemplo, aún suponiendo que las IPs fuesen públicas, imaginaos lo que sería si para decir una simple frase en un canal con 100 usuarios vuestra modesta conexión casera (56K, 256K, o poco más) tuviese que establecer una conexión con cada uno de los 100 usuarios para enviarles la frasecita de marras.

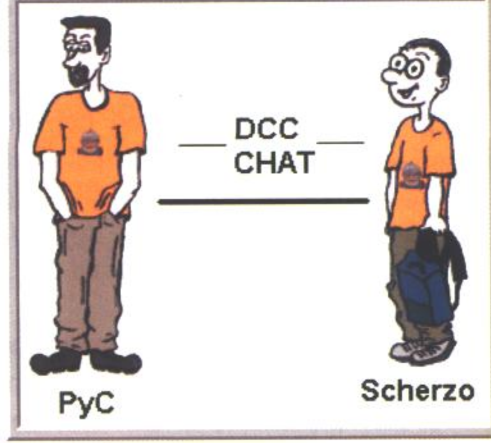
Volviendo al DCC, éste protocolo es el que permite realizar cualquier tarea que requiera una conexión punto a punto entre 2 usuarios. Las dos utilidades más sencillas que se nos pueden ocurrir en las que se necesite una conexión punto a punto son:

1- Enviar archivos: menuda locura sería si tuviésemos que enviar un archivo de 700MB a través del servidor de IRC mientras otros 3000 usuarios están enviando simultáneamente sus ISOs.

Chat privado: las queries, o chats privados entre dos usuarios de IRC, funcionan también a través del servidor de IRC (así que eso de privado...), pero hay varios motivos para que no queramos que nuestra conversación pase por un punto intermedio. Los motivos principales son dos: en primer lugar, la privacidad, y en segundo lugar, el no depender de las limitaciones técnicas del servidor intermedio.



Cuando hablo de limitaciones técnicas del servidor intermedio me refiero, ante todo, a dos problemas bien conocidos por todos los usuarios de IRC, que son: el **lag**,



y los **splits**. En un DCC Chat, es decir, un chat privado entre 2 usuarios mediante una conexión punto a punto, y no a través del servidor de IRC, el lag que haya depende tan sólo de la conexión de los dos usuarios, y no de la del servidor de IRC. Además, si ocurre un split en la red de IRC, o incluso si uno de los usuarios se cae, el DCC Chat sigue funcionando sin enterarse.



Lag y Splits... ...

Lag y Splits... ...

Split es cuando uno (o varios) servidores que forman parte de la RED IRC se "desconectan" temporalmente del resto de servidores que forman esa RED. Para no ocupar 2 páginas explicando los detalles, mejor te miras este enlace http://www.ayuda-irc.net/splits.shtml, está perfectamente expuesto con imágenes y todo :)

Lag es cuando por el motivo que sea se produce un retraso desde que escribes un mensaje hasta que el resto de usuarios lo reciben y viceversa (desde que otro usuario escribe un mensaje hasta que tu lo recibes). Las causas pueden ser muchas, desde que tu conexión deja de responder temporalmente hasta que el servidor de IRC al que estás conectado está saturado.

1.2. Establecimiento de la conexión

El establecimiento de la conexión en DCC es diferente a la del resto de protocolos que hemos visto hasta ahora en la serie RAW. Y es que no basta con hacer un telnet a un puerto determinado en el que hay escuchando un servidor, si no que tenemos que realizar unas "gestiones" previas para que ese puerto se abra en la máquina a la que nos queremos conectar.

Esto se debe a que el establecimiento de conexión se realiza a través del servidor de IRC. Recordemos que un usuario de IRC no tiene por qué conocer la IP de otro usuario, así que si queremos establecer una conexión punto a punto tendremos primero que "preguntarle" la IP del otro usuario al servidor. Para los que os hayáis frotado las manos pensando que esto se podría aprovechar para sacar IPs haciendo falsas consultas al servidor, o para los que os hayáis llevado las manos a la cabeza ante la inexactitud de lo que acabo de decir, ¡dejad todos las manos quietas! Lo he explicado así para que pilléis rápidamente el concepto, pero lo que ocurre en realidad no es que un usuario consulte la IP de otro usuario, si no justo todo lo contrario. Lo que hace el usuario es decirle al servidor: "dile mi IP a este usuario". Así nos "aseguramos" de que mediante DCC sólo se podrán conseguir las IPs de los usuarios que voluntariamente quieran. Pero aún sigo sin contar toda la verdad, ya que en realidad todo esto es mucho más divertido. Lo que decimos al servidor no es exactamente "dile mi IP a este usuario", si no "mi IP es ésta, anda, ve y díselo a este usuario". Por tanto, la IP que llega hasta un usuario con el que quieres establecer una conexión DCC, no es la IP que conoce el servidor, si no la que tú le digas... sea cual sea...

Una vez realizadas las gestiones previas (que consisten, resumiendo, en decirle al otro usuario nuestra IP y un puerto para que pueda conectarse), ya se podrá establecer una conexión TCP/IP de toda la vida que, por supuesto, se puede realizar mediante Telnet.

Como ya dijimos, existen básicamente 2 tipos de mensajes DCC (todos los demás los vamos a ignorar, por no estar tan extendidos):

- **DCC Send**
- **DCC Chat**

El **DCC Send** es una petición que dice al servidor IRC: "quiero enviar éste fichero a éste usuario".

El **DCC Chat** es una petición que dice al servidor

IRC: "quiero establecer un chat privado con éste usuario".

En ambos casos, hay que añadir la siguiente coletilla: "para ello, dile que se conecte a éste puerto en ésta IP". Es decir, el que lanza la petición de DCC es siempre el que actúa como **servidor** para la conexión TCP/IP que se establecerá, bien en el envío del archivo, o bien en el chat privado.

Para los que, al leer la "definición" de DCC Send, se hayan preguntado cómo funciona entonces un **fserver**, les aclaro que las peticiones que se hacen a un fserver no tienen nada que ver con las peticiones de DCC, si no que son simples comandos "parseados" por un script que automatiza el lanzamiento de peticiones DCC Send desde el fserver. Para los que no sepáis lo que es un fserver, no os preocupéis, que esto no tiene mucho que ver con el tema. :-)



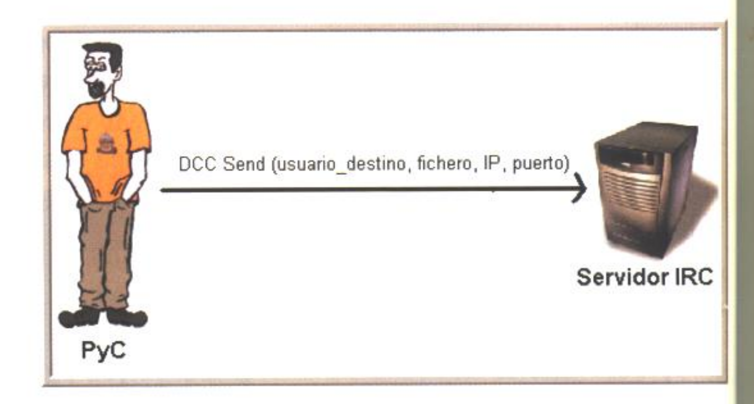
Fserver es un ...

Fserver es un servidor de ficheros pueden (o no) incluir los programas de CHAT y sirve para compartir tus archivos con otros usuarios, tienes sobre este tema mucha información en Internet, por ejemplo en http://www.readysoft.es/home/ihidalgo/intermedio/fileserver.html
Ya sabes, utiliza el mejor buscador de Internet: www.google.com

Para explicar mejor todo el proceso de conexión, voy a poner un ejemplo, en el cual el usuario PyC quiere enviar un archivo al usuario Scherzo.

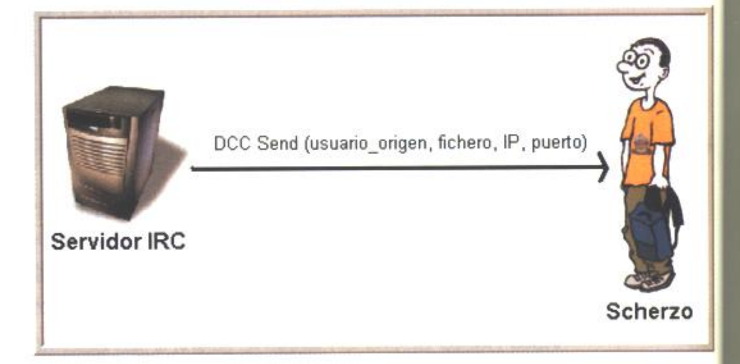
1.2.1. PyC lanza la petición al servidor de IRC

Tanto PyC como Scherzo están conectados al mismo servidor de IRC (condición imprescindible), por lo que PyC le lanza al servidor de IRC la consabida petición: "quiero enviar éste fichero a éste usuario. Para ello, dile que se conecte a éste puerto en ésta IP". (Por supuesto, más adelante veremos el formato real de esta petición XD).



1.2.2. El servidor de IRC devuelve la petición al usuario Scherzo

A continuación, el servidor de IRC simplemente retransmite esa petición a Scherzo: "Oye, que PyC me ha dicho que quiere enviarte éste archivo. Si quieres, conéctate a su IP, que es ésta, en éste puerto".



1.2.3. El usuario Scherzo acepta la petición

En este momento, Scherzo debe decidir si quiere recibir ese archivo de PyC. Para ello, en su aplicación cliente de IRC le aparecerá una ventana preguntándole si desea aceptar el DCC.

Si no acepta, aquí se acabó la historia. Pero vamos a suponer que si que acepta.

1.2.4. Scherzo establece la conexión TCP/IP con PyC

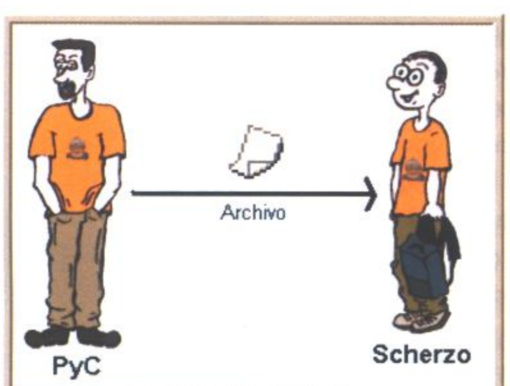
Una vez aceptada la petición, la aplicación cliente de IRC de Scherzo establecerá una conexión TCP/IP de las de toda la vida con la IP y puerto que se especificó en la petición. Si nuestro "cliente" de IRC es nuestro amado **Telnet**, podemos lanzar manualmente la conexión abriendo otro Telnet (aparte del que usamos para conectar al servidor de IRC), mediante:

telnet ip puerto

Donde ip es la ip que se nos especificó en la petición DCC, y puerto es el que se nos especificó en la petición DCC.

1.2.5. PyC envía el archivo a Scherzo a través de la conexión establecida

Pues eso, inmediatamente después de establecer la conexión, empezará a transmitirse el archivo a través de esa conexión establecida.



En el caso de que, en lugar de DCC Send, hubiese sido **DCC Chat**, llegados a este punto, en lugar de transmitirse el archivo, comenzaría una comunicación interactiva, donde cada usuario recibiría lo que el otro escribiese a través de esa conexión establecida.

1.3. Comandos RAW Para DCC

No hemos visto aún cómo son en realidad esas peticiones que el usuario lanza al servidor. Estas peticiones son, por supuesto, comandos RAW que forman parte del protocolo IRC, sobre el cual trataba mi anterior artículo.

Si recordamos algo sobre ese artículo, sabremos ya que los comandos que utilizamos en nuestro cliente de IRC no son directamente comprensibles por el servidor, ya que nuestro cliente hace un parseo y lo convierte al formato propio del servidor, que es el que llamamos formato RAW.

Como rápido recordatorio, si queréis hacer pruebas con los comandos RAW, podéis utilizar el comando **QUOTE** de vuestro cliente de IRC. Todo lo que pongáis después de un **/quote** será enviado en RAW al servidor.

El comando RAW más versátil es el PRIVMSG, así que podéis probar, por ejemplo, lo siguiente:

/quote privmsg PyC: hola, PyC! Como molan tus articulos!!! ;)

Si no utilizáis un cliente de IRC, si no un **Telnet** a pelo, os recuerdo que lo mismo se haría escribiendo:

PRIVMSG PyC: hola, PyC! Como molan tus articulos!!! ;)

El comando **PRIVMSG** no se usa sólo para hacer privados a otros usuarios, si no también para la conversación rutinaria en los canales. Por ejemplo con:

/quote privmsg #hackxcrack: buenas :D

Estaréis enviando un saludo al canal **#hackxcrack**. Por supuesto, para que este texto aparezca, tendréis que estar en el canal, o bien el canal tendrá que tener **modo –n**.

Recordemos también que el **CTCP** (Client To Client Protocol) funciona encapsulado en el comando **PRIVMSG**. Un ejemplo básico de **CTCP** es el siguiente:

/quote privmsg PyC:□VERSION□

Donde el símbolo ...

Donde el símbolo□ recordamos que es el ascii 1, el cual podemos escribir mediante la combinación de teclas: Alt + 1. Os recuerdo también que en algunas aplicaciones no funciona esta combinación de teclas, por lo que una solución es buscar una aplicación en la que si que funcione, y pastear el ascii desde ahí.

Pues ahora vamos a rizar el rizo un poco más. Si bien el **CTCP** no es más que un tipo de mensaje **PRIVMSG**, resulta que el **DCC** es a su vez un tipo de mensaje **CTCP**. Por lo que al final, resulta que el **DCC** también funciona mediante el famoso **PRIVMSG**. Bonito trabalenguas. :-)

Si consideramos que el protocolo **CTCP** comprende todos los comandos que se pueden ejecutar en un servidor de IRC que implican una comunicación entre 2 usuarios, podemos definir dentro de estos comandos un subconjunto, que es el formado por aquellos comandos que realizan una comunicación **directa** entre 2 usuarios estableciendo una conexión específica entre ambos. Este subconjunto es lo que llamamos DCC (**D**irect **C**lient to **C**lient).

Veamos ya un ejemplo de comando RAW para DCC, y luego lo explicaremos en detalle:

/quote privmsg PyC:□DCC CHAT chat 3645183495 4510□

¿Y esos numerajos? ¿No habíamos quedado en que lo que se enviaba en una petición era la IP y el puerto? Yo no veo la IP por ninguna parte...
Pues, aunque no la veas, ahí está, pero **codificada**. Y, por supuesto, yo os enseño ahora mismo cómo decodificarla. :-)

Codificación de IPs

¿Os apetece estudiar un poco de aritmética modular? Estoy seguro de que, dicho así, no os apetece lo más mínimo. XD
Pero es precisamente eso lo que debemos saber para codificar y descodificar las IPs en una petición de DCC.

Desde el punto de vista matemático, una dirección IP es un número de 4 cifras en base 256.

La base que utilizamos nosotros a diario es la base 10; por eso, cada cifra de un número esta comprendida entre 0 y 9. Por ejemplo, si queremos representar el número 256 en base 10, separando las cifras por puntos, sería ésta su representación: **2.5.6**.

Si queremos representar el mismo número en base 256, tenemos que tener en cuenta que cada cifra podrá valer ahora entre 0 y 255, y no sólo entre 0 y 9, así que ésta sería la representación del número 256 en base 256, separando las cifras por puntos: **1.0.** Curioso, ¿eh? El número 10 se representa como 1.0 en base 10, y el número 256 se representa como 1.0 en base 256. Esto ocurre con todas las bases. Si conocemos la base 2, también conocida como **binario**, sabremos que el número 2 se representa como 1.0 (supongo que conoceréis el clásico chiste que dice: existen 10 clases de personas, las que saben binario, y las que no). Si conocemos también la base **hexadecimal**, o base 16, sabremos que el 16 se representa como 1.0. Y así con todas las bases. :-)

Siguiendo con este curso intensivo de aritmética modular, veamos como ejemplo la representación del número 256 en binario: **100000000**, y en hexadecimal: **100.**

Para que podáis convertir cualquier número a cualquier base, os explico brevemente el

mecanismo. ¿Recordáis cuando estudiabais los números en el cole y os hablaban de centenas, decenas, y unidades? El número 256 se podría descomponer de la siguiente forma:

256 = 2*100 + 5*10 + 6*1

La cifra 2 es la de las centenas, el 5 la de las decenas, y el 6 la de las unidades. En realidad, los números que multiplican a cada cifra son simplemente potencias de la base, es decir, en este caso potencias de 10. Por ejemplo, $10^0 = 1$, $10^1 = 10$, $10^2 = 100$.

Aplicando la misma regla, la dirección IP **217.69.22.7** se podría representar como:
$\mathbf{217}*256^3 + \mathbf{69}*256^2 + \mathbf{22}*256^1 + \mathbf{7}*256^0$
Si realizamos esta suma nos sale, en decimal, el número **3645183495**.
¡Atiza! ¡Si es el mismo numerajo que salía en la petición de DCC que puse hace dos horas! (es que estoy perdiendo un poco el tiempo en IRC mientras escribo el artículo 0:-)

Por tanto, ésta es la fórmula que hay que utilizar para codificar una IP en una petición de DCC, cuya codificación no es más que la representación de la IP en base 10.

Decodificación de Ips

Para el proceso inverso, que es obtener una IP a partir de su codificación en base 10, no hay más que invertir la fórmula. Aquí es donde vemos por qué a todo esto se le llama aritmética modular. Y es que necesitamos operar con el **módulo** para ir obteniendo las 4 cifras que forman la IP.

En nuestro ejemplo, el número del que partimos es el 3645183495. Pues bien, empezamos dividiendo este número por el primero de los factores, que es 256^3, es decir:
$3645183495/256^3$ = **217'**...
Como vemos, no nos sale un número entero, pero la parte entera de ese número es **217**, que es precisamente la **primera cifra** de la IP.
Así que nos quedamos con ese 217 y seguimos operando, pero ahora con el **resto** de esa división. Para obtener el resto podemos aplicar directamente un operador de **módulo** si estamos programando en algún lenguaje, o con una buena calculadora. Por ejemplo, en el lenguaje **C** el operador de módulo es el **%** (**¡ojo!** Este % no tiene nada que ver con el que suele haber en las calculadoras). Si lo estamos haciendo "a mano" podemos obtener el módulo haciendo:

$217*256^3$ = 3640655872
3645183495 – 3640655872 = **4527623**

Que es lo mismo que se obtiene si hacemos:

3645183495 % (256*256*256) = **4527623**

A continuación, obtenemos la segunda cifra haciendo:
$4527623 / 256^2$ = **69'**...
Hallamos el nuevo resto:
4527623 % (256*256) = **5639**

Seguimos:
$5639 / 256^1$ = **22'**...
Hallamos el nuevo resto:
5639 % 256 = **7**

Y este último resto es precisamente la última cifra. :-)
Por tanto, la IP nos queda: **217.69.22.7**

Es muy fácil programar una pequeña aplicación que codifique y decodifique IPs, así que os lo propongo como ejercicio. ;-)

1.3.1. Comando RAW para DCC Chat

Os recuerdo el ejemplo:

/quote privmsg PyC:□DCC CHAT chat 3645183495 4510□

Ya sabemos qué es el **primer número**: la **IP codificada** en base 10.

El **segundo número** es el **puerto**, sin ningún tipo de codificación. Es decir, en el momento en que lanzamos esta petición de DCC, nuestra aplicación cliente de correo abrirá el puerto **4510** en espera de que el usuario PyC se conecte a él.

Si, en lugar de utilizar un cliente de IRC (como mIRC, kVirc, etc) estamos utilizando **Telnet** a pelo, la cosa es más complicada. Tendremos que abrir el puerto por medios más rústicos, por ejemplo utilizando **netcat**. Ya se explicó como abrir un puerto en escucha con netcat en el **número 8 de HackxCrack**, en el artículo sobre **Reverse Shell**.

Si estamos detrás de un **firewall**, es probable que tengamos problemas con DCC, ya que no es fácil abrir puertos dinámicamente detrás de un firewall. Para solucionar el problema, tenemos que abrir en el firewall unos puertos dedicados a DCC y, al mismo tiempo, configurar nuestro cliente de IRC para que utilice sólo esos puertos para DCC, en lugar de utilizar un rango aleatorio muy amplio, como suelen venir configurados por defecto (lo cual, por cierto, es más seguro, tal y como veremos más adelante).

Por ejemplo, en **mIRC**, nos vamos al menú File->Options->DCC->Options y ahí configuramos el rango de puertos donde pone DCC ports.

En el ejemplo de la imagen, se utilizarán los puertos **5100, 5101, y 5102** para lanzar las peticiones de DCC.

En segundo lugar, configuramos la **tabla NAT** del firewall para que abra esos puertos hacia nuestro PC. En la siguiente imagen está como ejemplo la tabla NAPT de un **router ADSL SpeedStream** para abrir los 3 puertos del ejemplo:

Configured NAPT Servers

Transport	Service	Server	Del
tcp	[illegible]	[illegible]	
tcp	21/ftp	192.168.2.2	
tcp	5102/5102	192.168.2.2	
tcp	5100/5100	192.168.2.2	
tcp	5101/5101	192.168.2.2	
tcp	113/113	192.168.2.2	

En esta tabla también está abierto el puerto 21 de FTP, y el 113 de IDENT, y otro que no os quiero enseñar :P, pero todo esto es otra historia...



Si utilizas Windows XP ...

Si utilizas Windows XP puede ser que tengas activado el Firewall que incorpora Windows, pues ya sabes, mejor lo desconectas (esto ya se enseñó en anteriores números). Si no sabes de qué estamos hablando pásate por nuestro foro (www.hackxcrack.com) y pregunta sobre el tema. Si utilizas un firewall por soft (por ejemplo el ZONE ALARM *www.zonelabs.com*) pues lo mismo, hay que configurarlo correctamente o desconectarlo temporalmente. Si no sabes como configurarlo pregunta en el foro y seguro que la peña te ayuda con el tema.

1.3.2. Comando RAW para DCC Send

Este es un ejemplo de petición para DCC Send:

/quote privmsg PyC :□DCC send "yonki.jpg" 1042580430 4511 56711□

Vemos que aparece el **nombre del archivo**

"yonki.jpg", y un numerajo más que en el DCC Chat. Los dos primeros numerajos son la **IP** y el **puerto**, exactamente igual que en DCC Chat. El **tercer numerajo** es simplemente el **tamaño del archivo** en Bytes.

Como ejercicio os propongo que decodifiquéis esta IP. ;-)

Pues bien... seguro que con todo esto que os he contado ya habrá aparecido en vuestras mentes pervertidas alguna idea maliciosa para sacar "provecho" del DCC pero, por si acaso, os voy a ahorrar el trabajo contando mis "ideas maliciosas".]:-)

2. INSEGURIDAD EN DCC

2.1. Pasos previos

Para poder llevar a cabo algunos de los experimentos que voy a contar a continuación, es necesaria una investigación previa para conseguir ciertos datos del usuario que usaremos (bonita redundancia), bajo su consentimiento, espero, para realizar los experimentos.

Para comprenderlo vamos a ver un escenario de ejemplo, en el cual hay 3 personajes:

Zer0Cul, Ac1dBrn, y PyC.

Zer0Cul y Ac1dBrn, como su nombre indica, son 2 representantes cualesquiera de la conocida especie de los lamers, mientras que PyC es un fornido y poderos.... esto... no, quiero decir que PyC no es más que un joven del montón, del grupo de los frikis. ;-)

Supongamos que Zer0Cul quiere hacer un DCC Chat a Ac1dBrn para urdir sus pérfidos planes de hackers. Así que lanza la petición de DCC Chat, ve cómo Ac1dBrn la acepta, y comienza el parloteo... Lo que no sabe Zer0Cul es que en realidad con quien esta chateando no es con Ac1dBrn, si no con PyC, que ha capturado la conexión. ;)

Ahora que ya os he puesto un poco en situación, y os he abierto el apetito, vamos a ponernos en el pellejo de PyC.

Para conseguir capturar esa conexión, PyC ha necesitado antes de nada recopilar una serie de datos acerca de Zer0Cul (el que ha lanzado la petición de DCC). La mejor forma de conseguir estos datos es mediante **ingeniería social** (ingsoc), una de las armas más poderosas. ¿Pero qué datos se necesitan? Pues los dos datos que circulan en una petición de DCC Chat, es decir: la **IP** de Zer0Cul, y el **puerto** que abre Zer0Cul para la petición de DCC Chat.

Cómo conseguir la IP de Zer0Cul

No puedo hacer aquí un tutorial completo de obtención de IPs, así que me limitaré a explicar lo más básico.

En primer lugar, si la red de IRC en la que estamos **no encripta** las IPs, como en el caso de EfNet, nos bastará con escribir en nuestro cliente de correo:

/dns Zer0Cul

En caso de que la red tenga **encriptación** de IPs, la forma más sencilla es conseguir establecer una conexión **DCC** con Zer0Cul mediante ingeniería social. Este método tiene la ventaja de que también nos servirá para el siguiente punto, que es la obtención del puerto. Aquí se pueden plantear 2 casos: que **consigamos que acepte una petición nuestra** de DCC, o que consigamos **que él mismo nos lance una petición** de DCC.

Para conseguir lo primero, muchas veces podemos aprovechar el **autoaccept** del cliente de correo de Zer0Cul. Ningún cliente autoacepta archivos **.EXE**, pero se puede probar con

archivos inofensivos, como **.TXT**. En cualquier caso, tantear el autoaccept es un riesgo ya que, si no acertamos, el tío ya estará alerta y será prácticamente imposible conseguir después que acepte un DCC mediante ingsoc. Así que recomiendo utilizar el autoaccept sólo cuando se sabe con seguridad qué tipo de archivos autoacepta.

Tanto si ha sido aprovechando el autoaccept, como si ha sido mediante ingsoc, una vez que está establecida la conexión de DCC tenemos que comprobar rápidamente las conexiones que tenemos establecidas. Para ello podemos utilizar el comando **netstat** desde una shell de Linux/Unix, o desde una ventana MS-DOS. Al ejecutar netstat veremos algo como esto:



Podemos ver 2 conexiones: la primera es la conexión que tenemos establecida con el servidor de IRC, por lo que no nos interesa ahora; en cambio, la segunda es precisamente la conexión de DCC que hemos establecido con Zer0Cul, por lo que ahí tenemos su IP visible (si, eso que he pixelado para que no veáis la IP de mi amigo :P). Por cierto, Poseidón es el nombre de la máquina en la que estoy ahora mismo. ¿A que mola poner nombres de dioses griegos a las máquinas de tu red local? ¡Incluso cumple con las recomendaciones del **RFC1178**! ;-)

Otra forma de comprobar la IP cuando nos acepta un DCC, sobre todo si sospechamos que no nos va a dar tiempo a hacer un netstat (bien porque enviamos un archivo muy pequeño, o porque no podemos estar pendientes del momento en que acepta) es tener un **sniffer** escuchando en nuestros puertos de DCC, y luego buscar los paquetes de esa conexión en el log del sniffer. Os recuerdo que en el primer artículo de la serie RAW expliqué cómo manejar un sniffer.

En el segundo caso, que es que sea él el que nos lance la petición de DCC, no hay ningún misterio, siempre y cuando utilicemos el software adecuado. Ni siquiera hace falta que aceptemos el DCC, ya que su IP se encontrará visible en la propia petición, antes de establecer ninguna conexión. El más famoso cliente de IRC, **mIRC**, no nos servirá para esta tarea, ya que no nos muestra los datos de una petición de DCC. Por tanto, tenemos las siguientes opciones:

-En **cualquier sistema**: utilizar un cliente de **Telnet**. :-)

En **Linux**: podéis utilizar **Kvirc** o **XChat**.

En **Windows**: podéis utilizar algún **script** sobre mIRC que sí que muestre las peticiones de DCC completas, como por ejemplo **IRCap**.

En **Windows, si no nos gusta utilizar scripts**: podemos tener un **sniffer** escuchando en el puerto de IRC (típicamente el **6667**) y cuando nos llegue la petición de DCC, que mIRC nos mostrará incompleta, ver el paquete completo en el log del sniffer.

Cuando nos llegue la petición veremos algo como esto en nuestra ventana de Status:

[18:38] -Zer0Cul- DCC Chat (62.36.131.206)
-
·· Zer0Cul H4x0r@DDzACj.AQs89f.virtual DCC CHAT chat 1042580430 4510 ··

Ahí podemos ver que la IP de Zer0Cul es 62.36.131.206 (como ejercicio os propongo que lo comprobéis decodificando la IP), y que el puerto que ha utilizado para la petición es

el 4510.

Cómo obtener los puertos de DCC de Zer0Cul

El tema de los puertos se puede complicar mucho más. El problema está en que no hay un puerto único para DCC, si no que normalmente se utiliza un puerto aleatorio dentro de un rango, que es precisamente el que os enseñé cómo configurar en mIRC. Nos encontramos aquí con algo sorprendente, y es que el DCC es mucho más inseguro si tenemos un **firewall**. Esto se debe a que, tal y como vimos anteriormente, si nuestro firewall no nos permite abrir puertos dinámicamente para DCC, no tendremos más remedio que abrir un determinado número de puertos en la tabla NAT del firewall. Este número suele ser pequeño, ya que nadie pierde el tiempo en abrir 200 puertos que no va a usar, además del peligro que eso conlleva. Por tanto, la gente que está detrás de un firewall (por ejemplo, los usuarios de ADSL con router en multipuesto) suele tener un número pequeño de puertos para DCC.

Así que nuestra tarea consiste en averiguar el rango de puertos que tiene configurado Zer0Cul para DCC. Para eso, tenemos que conseguir que nos envíe varios DCC (cuantos más, mejor) y hacer estadísticas de los puertos que vemos en cada petición.

Si no conseguimos estas estadísticas tendremos que utilizar en los pasos sucesivos alguna herramienta que muestree puertos por fuerza bruta pero esta solución, aparte de que nos secará los sockets en un periquete, muchas veces no será lo suficientemente rápida como para que nos de tiempo a capturar la conexión.

2.2. DCC Chat Hijacking

Esta técnica que voy a explicar se puede utilizar también para capturar conexiones de datos de FTP, tal y como veremos en el próximo artículo de la serie que, si todo va según lo previsto, tratará sobre FTP.

Es precisamente lo que hizo PyC en el ejemplo que os planteé hace un rato.

Antes de capturar conexiones ajenas, vamos a ver cómo podemos "capturar" nuestras propias conexiones.

Para ello, le pedimos a un amigo, por ejemplo Scherzo, que nos lance una petición de DCC Chat.

Si su petición es algo así:

·· Scherzo LCo@DDzACj.AQs89f.virtual DCC CHAT chat 1042580430 5510 ··

Lo primero que haremos será decodificar su IP que, en este caso, será 62.36.131.206. Ahora ya podemos lanzar el telnet:

telnet 62.36.131.206 5510

Con esto establecemos la conexión de DCC Chat con Scherzo. Para ver lo que escribimos nosotros mismos os recuerdo que tenéis que activar el **eco local** en vuestro cliente de Telnet. Y cuando nos escriba Scherzo... veremos que el texto que escribe Scherzo nos llega a pelo, y en ningún lugar aparece ninguna información sobre el **nick** o la **IP** de la persona que nos está hablando...]:-)

Supongamos que nos encontramos ahora en el escenario de ejemplo, y estamos en el pellejo de PyC. Ya hemos recopilado la información necesaria: IP de Zer0Cul, y puertos que utiliza para DCC. Ahora lo que nos hace falta es un tercer dato, que es realmente vital, y es saber **cuándo** lanzará Zer0Cul la petición de DCC Chat a Ac1dBrn.
Podríamos hacer un muestreo constante de los puertos en espera de que lance alguna petición,

pero esto, aparte de que cantaría en cualquier IDS, nos consumiría muchos recursos. Por tanto, es mejor que estemos pendientes de un parámetro que nos indica la actividad de nuestro querido Zer0Cul, que es su **IDLE**. :-) Es fácil programar un script que haga un muestreo del IDLE de un usuario y te avise cada vez que éste se pone a **0**, pero también podéis hacer el muestreo a mano. Y es que, a diferencia de las capturas en FTP, que requieren una gran velocidad, las capturas de DCC tienen la gran ventaja del **amplio margen de tiempo** disponible.

La razón de esto es que, a no ser que Ac1dBrn tenga **autoaccept**, habrá un margen de tiempo entre que reciba la petición de Zer0Cul hasta que la acepte. Este margen de tiempo muchas veces es de varios segundos, ya que es un factor humano, así que contamos con todo ese tiempo para colarnos nosotros antes que Ac1dBrn. :-)

Os recuerdo que para ver el **IDLE** de un usuario, si no estáis en el mismo servidor que ese usuario, basta con que hagáis:

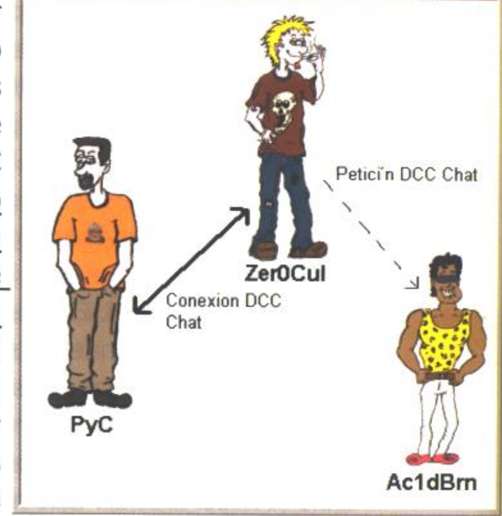
/whois Zer0Cul Zer0Cul

Es decir, un whois con su nick repetido. Si estáis en el mismo servidor, bastará con poner el nick una vez:

/whois Zer0Cul

Cada vez que Zer0Cul escriba algo en cualquier canal, o en cualquier privado, su idle se pondrá a 0. Pero también ocurrirá esto cuando lance una petición de DCC. Por tanto, podemos muestrear los puertos de DCC cada vez que se ponga a 0 su idle. Por supuesto, todo esto se puede automatizar mediante software, así que os propongo como ejercicio (para los que queráis sacar un 10) que hagáis una aplicación que automatice todo esto.

Evidentemente, si Zer0Cul es tan simpático de decir algo así como: "Oye, Ac1dBrn, te voy a enviar un DCC Chat, así que acéptalo" entonces no hace falta muestrear el idle ni más leches. XD



En cualquier caso, lo importante es que sepáis que el cliente de IRC de Zer0Cul no realiza ningún tipo de comprobación sobre la IP del que se conecta al DCC Chat por lo que, si sois más rápidos estableciendo la conexión que Ac1dBrn pulsando el botoncito de **ACCEPT**, entonces seréis vosotros los que ocupéis el puerto que tenía abierto Zer0Cul en espera de la conexión de Ac1dBrn.

Lo más interesante es que, tal y como vimos, en la conexión de DCC Chat se envía el texto a pelo, sin ninguna información sobre los **nicks** o las **IPs**. Por tanto, cuando en vuestro cliente de IRC aparece el nick del interlocutor en la ventana de DCC Chat, es porque el propio cliente pone el nick del interlocutor, asumiendo que éste será siempre el de aquel al que se lanzó la petición de DCC Chat.]:-)

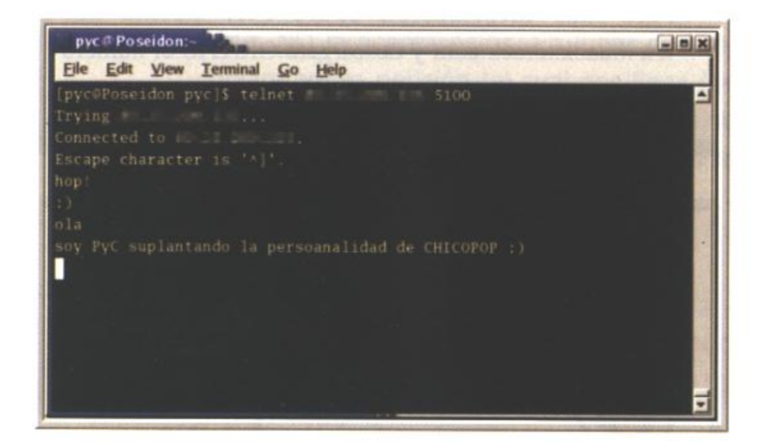
Chat with CHICOPOP
Waiting for acknowledgement...
DCC Chat connection established

Session Close: Wed Apr 30 15:56:56 2003

Session Start: Wed Apr 30 15:57:29 2003
Session Ident: CHICOPOP
Chat with CHICOPOP
Waiting for acknowledgement...
DCC Chat connection established

<CHICOPOP> hop!
<CHICOPOP> :)
> ola
<CHICOPOP> soy PyC suplantando la persoanalidad de CHICOPOP :)

Esto es lo que ve Zer0Cul en una conexión que intentó hacer con CHICOPOP, y que fue capturada por PyC.



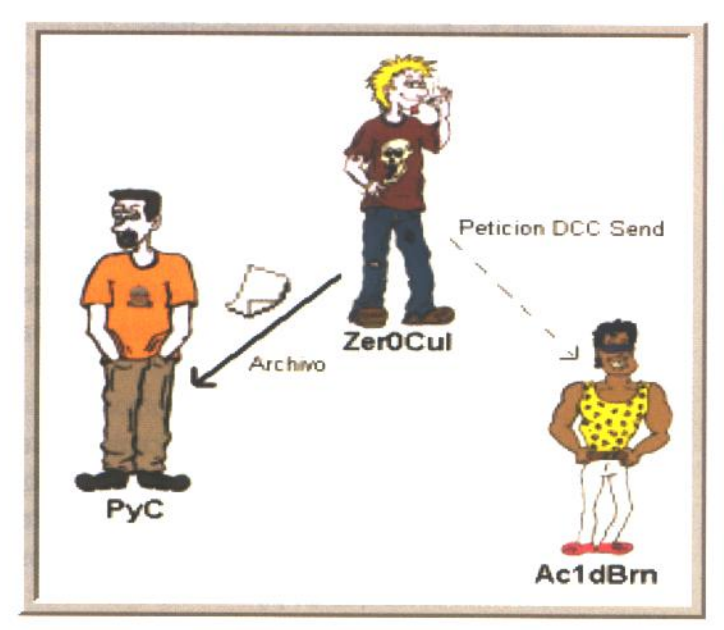
Y esto es lo que ve el propio PyC en la misma sesión, desde Telnet. Como podéis comprobar, no hay ninguna información sobre los nicks.

2.3. DCC Send Hijacking

La idea es exactamente la misma que con el DCC Chat, pero la diferencia es que aquí no se trata de una sesión interactiva, si no que inmediatamente después del establecimiento de la conexión comienza la transmisión del archivo.

Si hacéis la captura mediante un cliente de Telnet, recordad activar el **LOG** para luego poder ver el fichero. Bastará con que renombréis el archivo .log a la extensión adecuada. No podéis conocer la extensión original del archivo, ni su nombre, ya que la petición (en la cual iba incluido el nombre del archivo con su extensión) no os llegó a vosotros, si no al receptor "legítimo". Por tanto, tendréis que analizar el archivo para deducir su extensión.

Evidentemente, no me voy a poner aquí a hablar sobre las cabeceras de los distintos formatos de archivos, ya que se saldría bastante del tema, y esta revista sería más gorda que una biblia. :-)





Como puedes ver ...

Como puedes ver, en cada artículo te "proponemos" que investigues sobre temas que quizás nunca te has planteado. Busca información sobre las cabeceras de archivo y verás que cada tipo de archivo (.exe, .avi, .zip, .mp3 ...) está perfectamente definido gracias a su cabecera.

2.4. IP Spoofing en DCC

No hay que ser muy listo para darse cuenta de que spoofear la IP en una petición de DCC es absolutamente trivial. Si vuestro problema es que no sabéis lo que es el **IP spoofing**, os aclaro en un segundo que consiste simplemente en enviar, con algún fin, paquetes con una IP falsa, es decir, que no es vuestra IP real. El IP spoofing clásico consiste en modificar el campo de **IP de origen** en la cabecera de los datagramas IP, lo cual tiene muchos inconvenientes que no viene al caso comentar. Pero en este caso la modificación no se hace a nivel de red, si no dentro del propio protocolo DCC.

Veamos algunas ventajas que se pueden sacar del hecho de poner una IP que no es la nuestra en una petición de DCC.

2.4.1. A ver quién es más listo

Si nos paseamos a menudo por redes con encriptación de IPs, como el IRC-Hispano, o Netshock, nos podremos encontrar con algún listillo que intente utilizar con nosotros la técnica que expliqué antes para conseguir tu IP. Me refiero a conseguir por cualquier método (típicamente ingeniería social) establecer una conexión DCC contigo.

En el momento en que el listillo consiga que le envíes un archivo o le hagas un DCC Chat, estará convencido de que ya tiene tu IP. Claro que... lo que no sabe es que nosotros somos aún más listos que él. ;-)

Cuando sospechemos que alguien quiere conseguir nuestra IP intentando convencernos para establecer cualquier tipo de conexión DCC, podemos simular que hemos caído en la trampa, enviándole nosotros un DCC pero, por supuesto, esta petición de DCC no llevará nuestra IP, si no cualquier otra. La elección de la IP la dejo a vuestra imaginación, pero puede ser bastante divertido ver cómo el listillo comienza su "ataque" contra una IP de a guardia civil, o contra su propia IP.

2.4.2. FXP en DCC

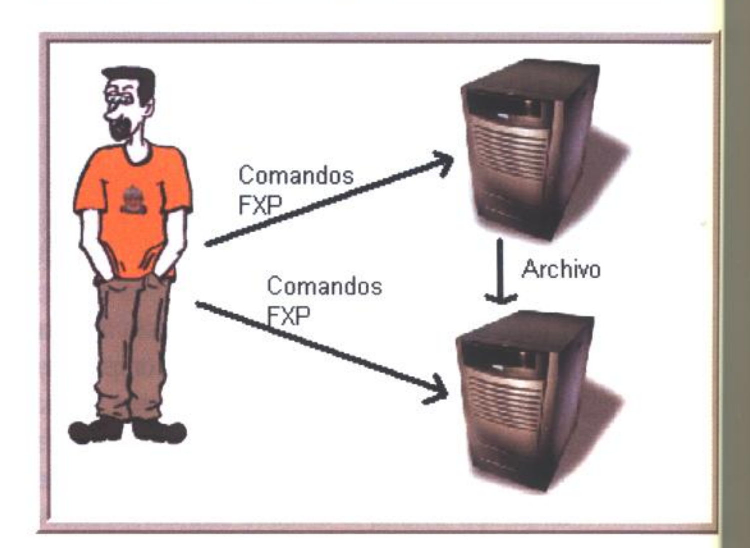
Bueno, lo llamo FXP por llamarlo de alguna forma. Espero que sepáis lo que es el FXP (**F**ile e**X**change **P**rotocol), pero os resumo contándoos que consiste en utilizar un cliente (típicamente de FTP, aunque en este caso será de DCC) para realizar una transferencia de archivos entre 2 servidores.

El FXP es el mecanismo utilizado por los couriers de los grupos de warez para distribuir las releases a través de los distintos servidores, sin necesidad de que ellos tengan conexiones de alta velocidad en su casita.



Para mas información ...

Para más información sobre FXP pásate por www.hackxcrack.com y descárgate GRATIS el número uno de esta revista en formato PDF ;)



¿Y cual es la utilidad de hacer FXP en DCC? Pues eso depende de vuestra imaginación, pero se me ocurren un par de ejemplos.

En primer lugar, si tienes una máquina con una conexión rápida (por ejemplo, en el curro, o en la facultad) y quieres bajar archivos de un Fserver rápido desde casa, puedes automatizar los downloads desde el Fserver hacia la otra máquina sin moverte de casa. Supongamos que el nick del Fserver es WrzServ, el nick de nuestra máquina remota es FastDld, y nuestro nick en casita es PyC. Vamos a verlo paso a paso:

paso 1: ponemos el archivo que queremos en la cola de download del Fserver

Se sale del tema del artículo explicar el funcionamiento de un Fserver, así que espero que ya conozcáis el clásico mecanismo de: **trigger** -> búsqueda del archivo mediante **dir** y **cd** -> **get**.

paso 2: nos llega el DCC send del Fserver

Cuando llega nuestro turno en la cola (queue) del Fserver, nos llegará su petición de DCC Send. NO la aceptaremos. En lugar de eso, copiaremos la petición tal cual nos llega (os recuerdo que para ver la petición completa no nos sirve mIRC, tal y como expliqué en el apartado 2.1).
La petición podría ser algo así:

:WrzServ!Wrz@82-15-31-37.uc.nombres.ttd.es PRIVMSG PyC :DCC SEND edlin.iso 1376722725 4510 681574400

paso 3: enviamos el DCC send a nuestra máquina remota

Esa petición que hemos copiado la reenviamos a FastDld, con sólo cambiar el nick de destino:

/quote privmsg FastDld :□DCC SEND edlin.iso 1376722725 4510 681574400□

paso 4: el autoaccept de FastDld chupa el archivo :)

Para que nuestra máquina remota pueda chupar los archivos que le "FXPeamos" tiene que tener **autoaccept** para ese tipo de archivos.

Mira que somos rebuscados, eh?... ¿pero y lo friki que resulta chupar archivos de esta forma tan complicada? :-)

Una segunda utilidad que se me ocurre para el FXP en DCC es el engañar simultáneamente a 2 listillos-busca-IPs, pero me parece a mí que eso mejor os lo cuento con DCC Chat, que es aún más divertido... ;-)

2.4.3. Puenteando un DCC Chat

Este truquillo no tiene mucha utilidad, excepto el poder reírte un rato o fardar de las chorradas que sabes hacer, pero a mi personalmente me encanta. :-)

Supongamos que hay 2 listillos que quieren conseguir nuestra IP, intentando establecer un DCC con nosotros. Para no variar, se llamarán Zer0Cul y Ac1dBrn. Ambos estarán deseando que les lancemos una petición de DCC, o bien que aceptemos una petición suya.

En el momento en que, por ejemplo, Zer0Cul nos tantee lanzándonos una petición de DCC chat estará la cosa hecha. Recibiremos algo como esto:

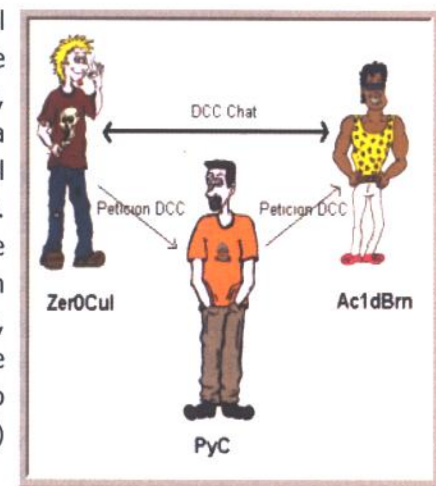
:Zer0Cul!H4x0r@82-15-31-37.uc.nombres.ttd.es PRIVMSG PyC :DCC CHAT chat 1376722725 4510

Así que de inmediato modificamos ligeramente esa petición, para lanzar esto:

/quote privmsg Ac1dBrn :□DCC CHAT chat 1376722725 4510□

Es decir, mandamos exactamente la misma petición a Ac1dBrn.

Desde el punto de vista de Ac1dBrn, TU serás el que le estará lanzando la petición de DCC Chat, cuando en realidad esa era la petición que te lanzó a ti Zer0Cul. En el momento en que Ac1dBrn la acepte, desde el punto de vista de Zer0Cul TU serás el que lo habrá aceptado. A partir de ese momento comenzarán un diálogo de besugos, creyendo cada uno de ellos que el otro interlocutor eres tú. 0:)

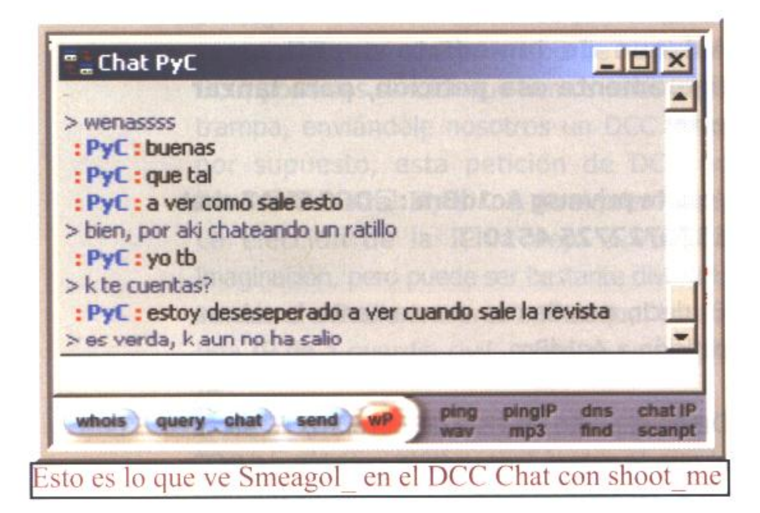


Aquí os copio un par de capturas de pantalla de un DCC chat puenteado entre 2 colaboradores voluntarios, para que veáis que desde el punto de vista de ambos es con PyC con quien creen estar hablando.

Chat PyC

<PyC> wenassss
<shoot_me> buenas
<shoot_me> que tal
<shoot_me> a ver como sale esto
<PyC> bien, por aki chateando un ratillo
<shoot_me> yo tb
<PyC> k te cuentas?
<shoot_me> estoy deseseperado a ver cuando sale la revista

Esto es lo que ve shoot_me en el DCC Chat con Smeagol_



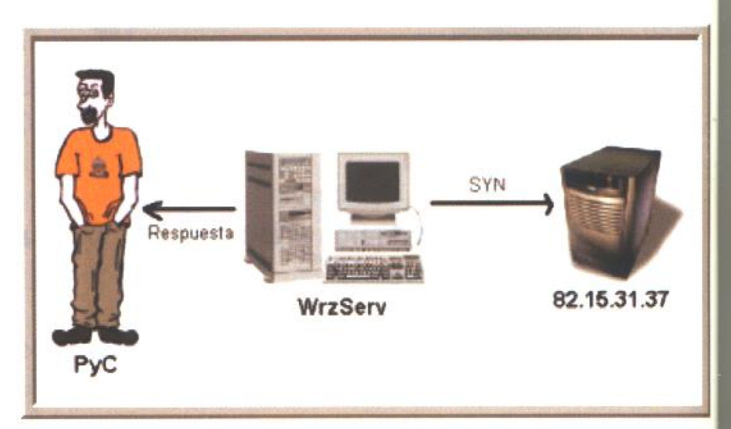
Esto es lo que ve Smeagol_ en el DCC Chat con shoot_me

Os dejo como <u>ejercicio</u> que apliquéis esta misma idea a DCC Send en lugar de a DCC Chat, tal y como os expliqué en el apartado anterior sobre FXP.

2.4.4. Escaneo anónimo de puertos a través de un Fserver

No me cabe duda de que vais a flipar con lo increíblemente retorcida que es mi mente, ya que hay miles de formas más sencillas de hacer un escaneo de puertos que la que voy a plantear aquí, pero os recuerdo que mi fin no es enseñar a nadie a ejecutar un programa con un botón muy grande que ponga DESTRUIR MAQUINA REMOTA, si no el hurgar en las tripas de los protocolos hasta que nuestra ansia de conocimiento se sacie al conocer hasta los más rebuscados trucos.

Aquí os explicaré como hacer un escaneo de puertos anónimo, utilizando un Fserver como **bouncer**. Los resultados son bastante pésimos, os lo advierto. xD



El sistema de escaneo de puertos más típico es el de lanzar **intentos de conexión** a cada puerto, y ver cual es la respuesta. No puedo entrar aquí en el tema de los flags de la cabecera TCP por falta de espacio pero, para los que sepáis algo del tema, me refiero al envío de paquetes con el flag **SYN**. Cada vez que se intenta establecer una conexión TCP/IP, lo primero que hace el cliente es enviar al servidor un paquete con el flag SYN. Esto, por supuesto, también ocurre al intentar establecer una conexión de DCC.

Planteamos una nueva situación. Tenemos un Fserver, cuyo nick es WrzServ, y una máquina que queremos escanear, cuya IP es 82.15.31.37, y nosotros mismos somos el usuario PyC, para no variar. :-)

La condición necesaria para que esto funcione es que el Fserve **autoacepte** algún tipo de archivo, como ocurre muchas veces. Yo sólo realicé experimentos con un Fserve que utilizaba **Ircap 6.999**. Propongo como <u>ejercicio</u>

que intentéis conseguir esto mismo con otros scripts, para lo cual tendréis que hacer una labor de investigación para analizar las respuestas que os pueda dar el script, tanto ante un puerto abierto, como ante un puerto cerrado. Suponemos en el ejemplo que WrzServ utiliza Ircap 6.999, para que veáis las respuestas que da este script ante un "escaneo".

1: envío de la petición spoofeada

Tenemos que construir una petición de **DCC Send** (ya que DCC Chat es improbable que sea autoaceptado) en la que la **IP** sea la de la máquina que queremos escanear (82.15.31.37), y el **puerto** pues... el que queremos escanear. :-)

Vamos a ver si en esa IP hay un servidor de SMTP corriendo (puerto 25), así que construimos el siguiente paquete:

/quote privmsg WrzServ :□DCC SEND "uber.lco" 1376722725 25 6922□

Por supuesto, el primer número es la codificación de la IP que queremos escanear, el segundo es el puerto (25 = SMTP), y el tercero un tamaño de archivo que nos inventamos, al igual que el nombre del archivo.

2: respuesta del script ante un puerto abierto

Si hay un servidor SMTP en la máquina que estamos escaneando, la respuesta de WrzServ será algo como esto:

-WrzServ- Recibido caca.pis (89 bytes). Tus créditos en mi Fserve son de 300000 (+ 0)

Como vemos, el Fserver nos dice que ha recibido 89 bytes, lo cual significa, no sólo que el puerto le ha **respondido** cuando ha lanzado la conexión, si no que además le ha enviado un **mensaje de bienvenida** de 89 bytes. En el caso de que no haya mensaje de bienvenida, nos responderá con **(0 bytes)**. La respuesta suele tardar bastante en llegar, os aviso.

3: respuesta del script ante un puerto cerrado

En este caso, la respuesta puede ser algo como esto:

-WrzServ- Recibido caca.pis (bytes). Tus créditos en mi Fserve son de 300000 (+ 0)

Como vemos, en este caso no ha recibido nada, ya que no ha podido conectar con el puerto. La diferencia con el caso de que el puerto esté abierto pero sin mensaje de bienvenida, es que en ese caso nos pondría (**0 bytes)**, en lugar de poner simplemente (**bytes)**.

Por supuesto, la máquina que está siendo escaneada no conocerá en ningún momento vuestra IP, ya que será WrzServ el que se esté conectando a sus puertos.

2.5. Desvariando de todas las formas posibles

Si usamos nuestra retorcida imaginación, se nos pueden ocurrir mil virguerías más, la mayoría sin la más mínima utilidad. Por ejemplo, si lanzamos esta petición a un usuario cualquiera, por ejemplo a Zer0Cul:

/quote privmsg Zer0Cul:□DCC CHAT chat 3586965548 23□

Cuando acepte la petición, su cliente de IRC se conectará, sin que él lo sepa, a un servidor en el que unos frikis han puesto la peli de **star wars** en ascii. Todos los fotogramas de la peli le irán llegando a través del DCC Chat. Realmente inútil, ¿verdad? :)

Otra idea estúpida consiste en lanzar una petición de DCC Chat donde la ip esté spoofeada para que sea la de alguna máquina que tenga un servidor de **echo** (puerto 7) corriendo. Algunos cablemodems, por ejemplo, tienen servicio de eco. Cuando acepten el DCC Chat, todo lo que escriban en la sesión de chat les será devuelto como si estuviesen hablando con un loro.

Otra cosa que podéis hacer es conectaros a cualquier tipo de servidor utilizando DCC Chat. Por ejemplo, servidores de FTP, POP3, SMTP, WEB, etc, etc. Podéis escribir comandos y recibir las respuestas como si se tratase de Telnet. Y luego dicen de emacs, pero se puede hacer todo sin salir de tu cliente de IRC, así que, como sigamos así, mIRC terminará siendo el sistema operativo del futuro. xD

Autor: PyC (LCo)
Ilustraciones: MariAn (LCo)
Agradecimientos: Scherzo (LCo), uhm, CabRa, CHICOPOP, shoot_me, kaiszz, Smeagol_, ...

CURSO DE VISUAL BASIC (V): ACCESO A DATOS II, ALTAS, BAJAS Y MODIFICACIONES

DEDICADO A OSCAR ESQUINAS SAEZ (XEVIAN), FALLECIDO EL 28/04/2003

POR PEDRO DEL VALLE

Esta es la quinta entrega del Curso de Visual Basic y la segunda dedicada al Acceso a Datos. El trabajo con Datos es, en la actualidad, uno de los valores más apreciado por las empresas.

No importa cuántos datos seas capaz de almacenar, lo imprescindible es poder acceder a ellos cuando los necesitas !!!

- Bienvenidos de nuevo. En el último artículo dimos los primeros pasos en el mundo del acceso a datos. En el ejercicio propuesto, creamos un formulario con 4 cajas de texto que mostraban los datos de la BD, y cuatro botones para movernos por ellos.

Al final del ejercicio os comenté que si llegábamos al principio o final de los datos, y volvíamos a pulsar anterior o siguiente (dependiendo de donde estemos posicionados), el programa devolvería un error. Os comenté que las propiedades que debíamos usar para reparar esto eran EOF (End Of File) y BOF (Begin Of File). Estas dos propiedades nos devuelven verdadero si el Recordset está posicionado al final o principio del fichero respectivamente.

Para no darle muchas vueltas al tema, porque seguro que muchos ya lo habréis solucionado, os escribo el código necesario para depurar este error

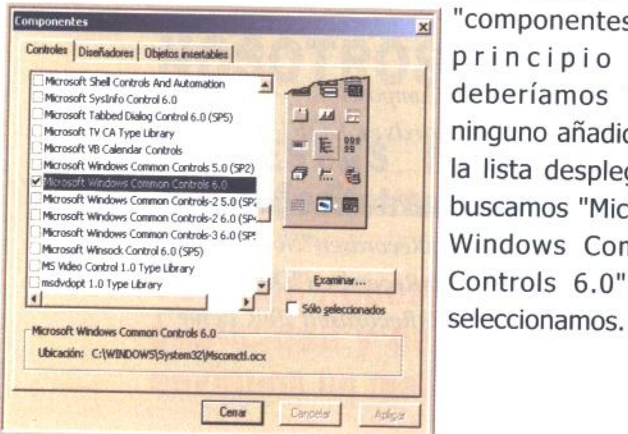
```
Private Sub CmdAnterior_Click()
  If Not RsRecordset.BOF Then
    RsRecordset.MovePrevious
    LlenarCampos
  End If
  If RsRecordset.BOF Then
    RsRecordset.MoveLast
    LlenarCampos
  End If
End Sub

Private Sub CmdSiguiente_Click()
  If Not RsRecordset.EOF Then
    RsRecordset.MoveNext
    LlenarCampos
  End If
  If RsRecordset.EOF Then
    RsRecordset.MoveFirst
    LlenarCampos
  End If
End Sub

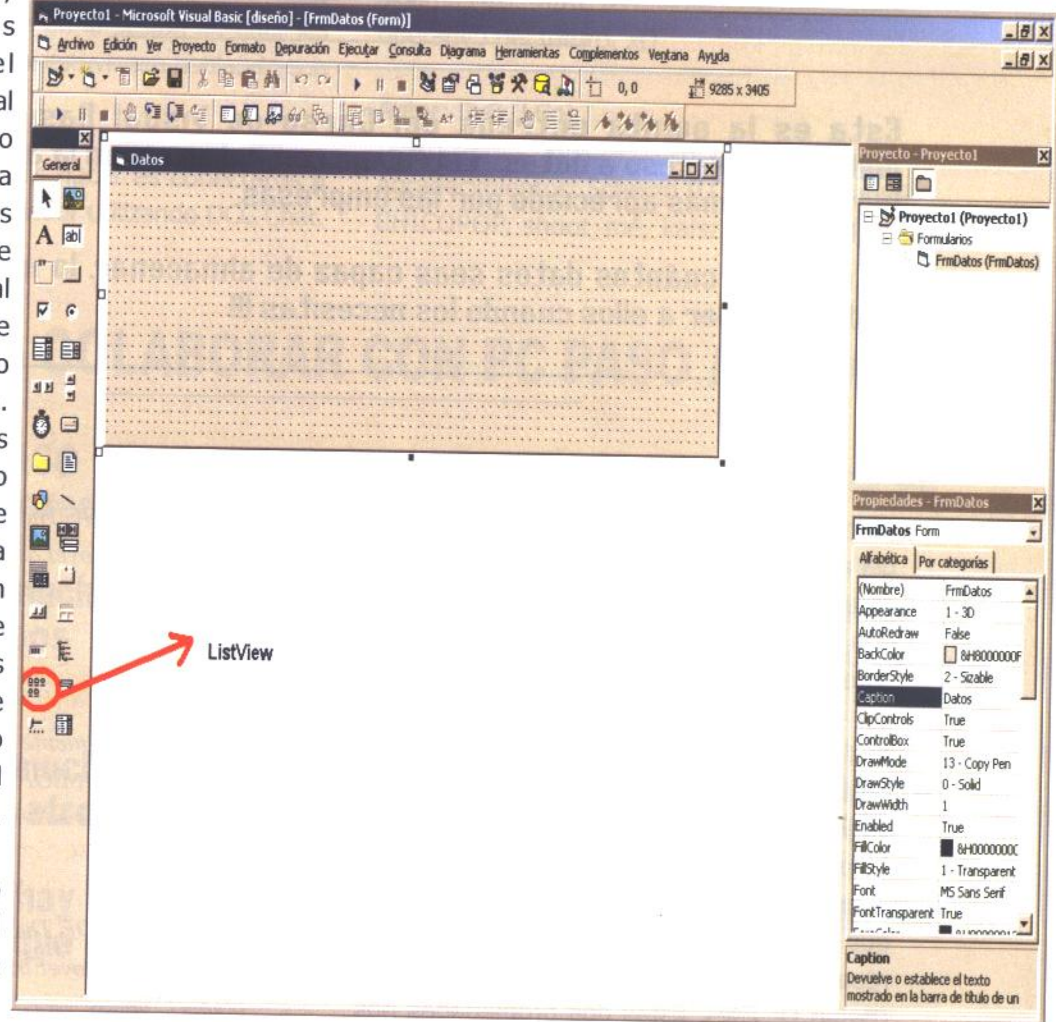
Public Function LlenarCampos()
      If Not RsRecordset.EOF And Not RsRecordset.BOF Then
        TxtIdCliente = RsRecordset("IdCliente")
        TxtNomCliente = RsRecordset("NomCliente")
        TxtDirCliente = RsRecordset("DirCliente")
        TxtTelCliente = RsRecordset("TelCliente")
  End If
End Function
```

¿Qué hemos hecho?, pues, bajo mi criterio, le estoy diciendo al programa que cuando estemos al final de fichero (o principio) y se vuelva a pulsar "Siguiente" (o "Anterior") se de la vuelta entera. Es decir, si estamos en el último registro del Recordset y pulsamos "Siguiente", el programa volverá al primer registro y lo mostrará. Otra posibilidad es mostrar un mensaje cuando estemos al final o principio de fichero, lo dejo bajo vuestro criterio. Bien, empecemos entonces con algo nuevo. Lo que vamos a hacer para empezar es un formulario donde se muestren todos los registros de la base de datos. Para ello utilizaremos el objeto "ListView".

Abrimos un nuevo proyecto de Visual Basic, y elegimos EXE estándar. Vamos al menú "Proyectos" y seleccionamos "componentes". En principio no deberíamos tener ninguno añadido. En la lista desplegable, buscamos "Microsoft Windows Common Controls 6.0" y lo seleccionamos.



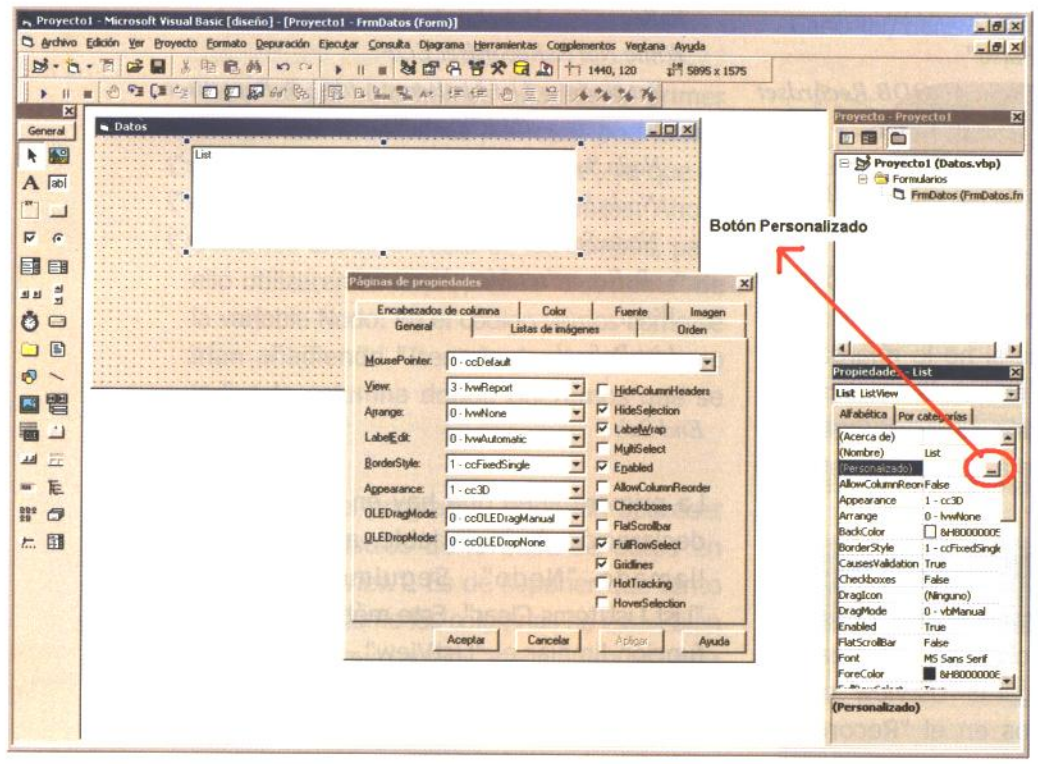
Al aceptar veremos que se nos han añadido varios objetos en la barra de herramientas de la izquierda. El que a nosotros nos interesa ahora mismo es el "ListView", así que agreguémoslo a nuestro formulario



Recordad que...

Recordad que debemos agregar las referencias necesarias para el acceso a datos. Estas se encuentran en el menú "Proyectos->Referencias". Para agregarlas tenemos que seleccionar "Microsoft ActiveX Data Objects 2.6 Library"

Pasamos a configurar el "ListView". Para ello debemos pulsar en el botón "Personalizado" que está en el cuadro de propiedades, apareciéndonos la hoja de propiedades del objeto.



En este objeto queremos que aparezcan los datos de la BD, por lo tanto, deberíamos agregar tantas columnas como campos hay en la base de datos. Si nos fijamos, en la hoja de propiedades tenemos una pestaña con el literal "Encabezados de columna". Vamos a ella, y comenzamos a añadir encabezados, picando en el botón "insertar columnas".

Aquí lo único que vamos a indicarle es el nombre. Para no complicarlo, le pondremos los mismos que a los campos de la base de datos.
La primera columna se llamará "IdCliente", igual que el primer campo. Cuando lo escribamos, volvemos a pulsar "Insertar Columna" y añadimos todas las demás.

Volvemos a la pestaña "General", y cambiamos la propiedad "View" a "lvwReport", para visualizar los datos en línea. También, y para recrear una buena visualización de los registros, seleccionamos, en la misma hoja de propiedades, "FullRowSelect" (selección completa de la línea cuando pulsemos en ella) y "Gridlines" (líneas separadoras entre registros).

Pulsamos "Aceptar" y ejecutamos el programa. De primeras, si hemos hecho todo bien, debería aparecernos el formulario, con el "ListView" vacío, los encabezados de columna escritos y las líneas que separarán los registros.

¿Todo bien?, pues rellenemos el List. Vamos al código, y declaramos dos variables, un "Recordset" y un "Connection"

```
Option Explicit
Dim Conn As ADODB.Connection
Dim RsRecordset As ADODB.Recordset
```

Lo que viene ahora es complicado. Para hacernos una idea, lo que tenemos que conseguir es que se añadan todos los registros de la base de datos en el "ListView". Para ello tenemos que cargar el "Recordset" con estos registros, recorrerlo por completo e ir añadiéndolos al "ListView". Vamos a crear una rutina, que llenará el Objeto. Pongámosle un nombre significativo, como por ejemplo, "LlenarList". Antes de codificarla deberíamos conectar con la base de datos. Vamos al "Form_Load" y conectamos como lo hicimos en el ejercicio anterior.

```
Private Sub Form_Load()
    Set Conn = New ADODB.Connection
        Conn.Open "Provider
        =Microsoft.Jet.OLEDB.4.0;Data
        Source =" & App.Path &
```

```
                "\bd.mdb;Persist Security
                Info=False"
    Set RsRecordset = New ADODB.Recordset
    RsRecordset.Open "Clientes", Conn,
        adOpenDynamic, adLockOptimistic
End Sub
```

Acordaos que debéis guardar el proyecto progresivamente, no solo porque podéis perder los datos, sino porque si no lo guardáis, el objeto "App.Path" hace referencia a una carpeta temporal de la instalación de Visual Basic, por lo que no nos encontrará la base de datos de Access.

Probamos la conexión con ctr + F5, para arreglar posibles errores antes de seguir complicando el ejercicio. Una vez creada la conexión, vamos a cargar el "ListView" con los datos que ya tenemos en el "Recordset".

Codifiquemos la rutina "LlenarList". Para ello debemos hacer un bucle que recorra el "Recordset", desde el BOF hasta el EOF. Lo que tenemos que hacer primero es posicionarnos al principio, con "RsRecordset.MoveFirst". Después de colocarnos, generamos un bucle repetitivo mientras que no lleguemos a final de fichero. Dentro de este bucle llenaremos el "ListView".

Esto puede parecer complicado, ya que para llenar el List tenemos que declara una variable de tipo "ListItem". Yo la he declarado en la misma rutina "LlenarGrid", pero se puede declarar igualmente debajo del "Option Explicit". Esta variable será la intermediaria entre la el "Recordset" y el "ListView". Como que la variable "ListItem" es un objeto, debemos instanciarla con el comando "Set". Mejor pongo el código necesario y lo comentamos:

```
Sub LlenarList()
  Dim Nodo As ListItem
  List.ListItems.Clear
  RsRecordset.Requery
  RsRecordset.MoveFirst
  While Not RsRecordset.EOF
    Set Nodo = List.ListItems.Add(, , RsRecordset
            ("IdCliente"))
    Nodo.SubItems(1) = RsRecordset("NomCliente")
    Nodo.SubItems(2) = RsRecordset("TelCliente")
    Nodo.SubItems(3) = RsRecordset("DirCliente")
    RsRecordset.MoveNext
  Wend
  List.Refresh

End Sub
```

La primera línea que hay en la rutina es la declaración de la variable, a la cual yo he llamado "Nodo". Seguimos, y vemos "List.ListItems.Clear". Este método tiene como función limpiar el "ListView". Posteriormente refrescamos el "Recordset", con su propiedad "Requery" y nos posicionamos justamente al principio de los datos. Hemos hecho todo esto porque estamos pensando que en un futuro haremos actualizaciones contra la base de datos, y por lo tanto necesitaremos borrar el List, refrescar el Recordset (para mostrar los cambios) y movernos al primero.
Ahora viene el bucle. Le estamos diciendo al programa que mientras no lleguemos al final del fichero, sigue repitiendo la operación (While Not RsRecordset.EOF).
Ahora viene tal vez lo más complicado. Se trata de cargar la variable Nodo con los datos que queremos y a parte, pasárselas al "ListView". Para ello tenemos este método:

```
Set Nodo = List.ListItems.Add(, ,
RsRecordset("IdCliente"))
```

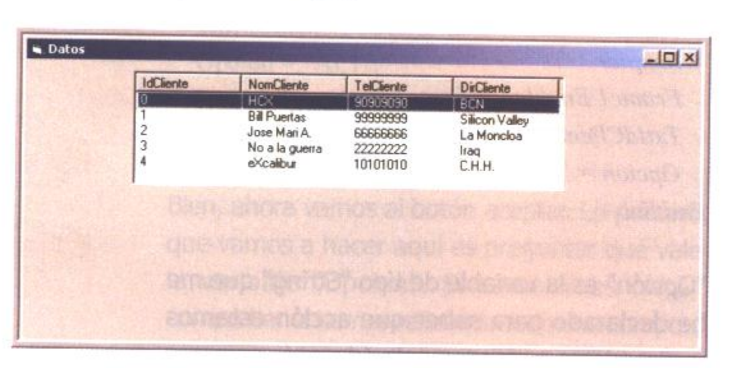
Aquí, lo que estamos haciendo es decirle a la variable Nodo, que se le añada el contenido del campo "IdCliente" del primer registro, y a su vez, añadiéndola al "ListView". Si nos fijamos, hemos omitido los dos primeros parámetros del método "List.ListItems.Add", ya que no los consideramos necesarios, pero podrían ser añadidos, ya que son el índice y la clave de

este "Item".

Hasta ahora solo hemos añadido el primer campo del "Recordset", el cual podríamos considerar como el campo "Madre", ya que es el campo clave de la base de datos. Ahora debemos añadir el resto de los campos, para ello utilizaremos la propiedad "SubItems" de la variable Nodo. En el código vemos como se irían añadiendo "Items", donde el número indica la columna donde queremos que se muestre.

Ejecutamos el programa. Deberían aparecer todo los registros de la base de datos en nuestro "ListView". Es de suponer que tanto el "ListView" como las columnas no están bien colocados. Seguro que os aparece una columna en blanco mucho mas grande que las demás. Hagamos entonces el "ListView" del tamaño necesario (seguramente tendréis que reducirlo) o bien hacer las columnas más grandes en su hoja de propiedades.

Bueno, finalmente nos debería quedar algo parecido a esto



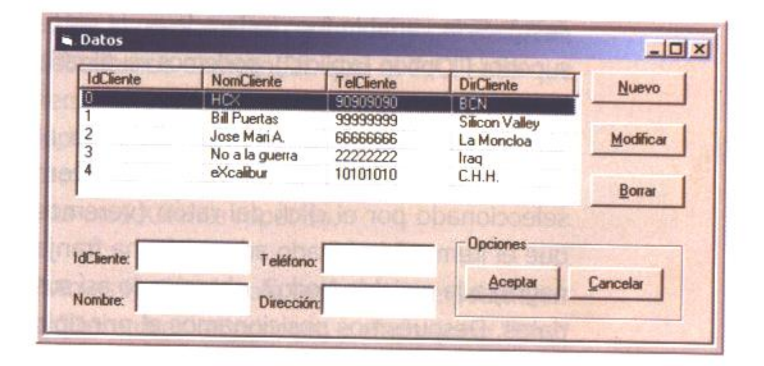
Llegados a este punto, creo que estamos listos para hacer un mantenimiento completo. Vamos a incorporar cajas de texto que nos ayudarán a añadir y modificar registros. Podéis utilizar las mismas que en el ejercicio anterior. También agregaremos unos botones que nos permitirán Añadir, Modificar y Eliminar registros. Por ejemplo, bajo mi criterio, he añadido tres botones (Nuevo, Modificar, Borrar) a la derecha del "ListView", y abajo, los botones Aceptar y Cancelar, dentro de un Frame.



Los frames son ...

Los frames son objetos contenedores, es decir, si tenemos dos botones dentro de un frame y deshabilitamos ese frame, automáticamente se deshabilitan los botones. Si esto no ocurre es porque no tenemos los botones dentro del frame. Lo podemos solucionar, poniéndonos encima del botón, cortándolo, y pegándolo en el frame

Así es como quedaría:



Estupendo, esto ya va cobrando forma. Sería interesante que cuando el usuario de esta aplicación picase sobre alguno de los registros, los datos de este pasaran a las cajas de texto. Es una operación sencilla, a parte, si hacemos esto, estaremos posicionando el "Recordset" en el registro seleccionado. Creemos una rutina que traspase los datos a los "TextBox". Por ejemplo, la podemos llamar "Pintar".

A esta rutina la debemos llamar desde el evento click del "ListView", así, cuando pulsemos sobre un registro, los datos pasarán a las cajas de texto.

```
Sub Pintar()
Dim Nodo2 As ListItem
   Set Nodo2 = List.SelectedItem
   RsRecordset.MoveFirst
     RsRecordset.Find "IdCliente=" & Nodo2
   If Not RsRecordset.EOF Then
     TxtIdCliente.Text = RsRecordset("IdCliente")
```

```
            TxtNomCliente.Text =
            RsRecordset("NomCliente")
            TxtTelCliente.Text =
            RsRecordset("TelCliente")
            TxtDirCliente.Text =
            RsRecordset("DirCliente")
    End If
End Sub
```

Comentémosla un poco.
Vemos que se ha declarado otra variable de tipo "ListItem" (Nodo2). Esto debe hacerse en los casos que la variable Nodo fue declarada dentro de la rutina LlenarList (como es en mi caso). Si la variable fue declarada en la parte superior ("Option Explicit"), podemos reutilizarla de nuevo. Al hacer "Set Nodo2 = List.SelectedItem" le estamos diciendo al programa que meta el contenido del ítem seleccionado por el click del ratón (veremos que el ítem seleccionado adquirirá una franja negra) a la variable Nodo2, obteniendo así sus datos. Después nos posicionamos al principio del "Recordset" y buscamos el primer registro que cumpla la condición explícita en la propiedad "Find". Esta propiedad busca el primer dato que cumpla con su condición, es decir, en este caso buscará el primer registro que tenga como IdCliente el número del ítem seleccionado en el "ListView".



Hay que tener ...

Hay que tener en cuenta que el "Recordset" y el "ListView" son dos objetos totalmente independientes, es decir, el que estemos haciendo click en un registro del List no quiere decir que automáticamente el "Recordset" se posicione en este, sino que tenemos que hacerlo nosotros por código.

Ejecutamos el programa y probamos que funcione.

¿Qué tal?, ¿va bien?, perfecto!!, pues añadamos ahora nuevos registros. Primero haremos un método que resulta bastante atractivo. En el "Form_Load" deshabilitaremos los botones Aceptar y Cancelar, ya que estos dos solo podrán ser utilizados cuando se haya propuesto antes una de las acciones (Nuevo, Modificación, Baja).

Por ejemplo, para deshabilitarlos, yo lo que haré es poner la propiedad ".Enabled" del "Frame1" a False. Como ya he dicho, esto lo podemos escribir en el "Form_Load" o directamente en el cuadro de propiedades, en tiempo de diseño. Ahora que hemos prohibido su utilización, codificaremos el botón Nuevo. Debemos declararnos una variable de tipo "String" que sea general al formulario (declararla en la parte superior, en el "Option Explicit").

Esta variable nos dirá la acción que estamos haciendo, es decir, cuando pulsemos el botón Nuevo, la variable se cargará con el valor "ALTA", cuando modifiquemos, su valor será "EDITAR", y cuando borremos, su valor será "BAJA". También debemos limpiar las cajas de texto, para permitir escribir, y colocar el foco en el primer "TextBox".

```
Private Sub CmdNuevo_Click()
    Limpiar
    Frame1.Enabled = True
    TxtIdCliente.SetFocus
    Opcion = "ALTA"
End Sub
```

"Opción" es la variable de tipo "String" que me he declarado para saber que acción estamos realizando en cada momento (después veremos su verdadera utilidad).

Para ser más pulcros, ahora deberíamos deshabilitar los botones Nuevo, Modificar y Baja, ya que no deberían tocarse hasta que se decidiese cancelar o aceptar la operación (y creedme que los usuarios lo tocarán), así que os propongo que hagáis un par de rutinas para habilitar y deshabilitar los botones.

```
Private Sub Habilitar()
   CmdNuevo.Enabled = True
   CmdModificar.Enabled = True
   CmdBorrar.Enabled = True
End Sub
Private Sub Deshabilitar()
   CmdNuevo.Enabled = False
   CmdModificar.Enabled = False
   CmdBorrar.Enabled = False
End Sub
```



Es muy importante ...

Es muy importante utilizar rutinas en aquellas acciones que creamos que se van a repetir varias veces, ya que ahorramos cantidad de código y se hace más legible.

También llamaremos a la rutina Deshabilitar cuando pulsemos Nuevo. El código necesario para codificar el botón sería el siguiente.

```
Private Sub CmdNuevo_Click()
   Limpiar
   Frame1.Enabled = True
   TxtIdCliente.SetFocus
   Opcion = "ALTA"
   Deshabilitar
End Sub
```

Bien, ahora vamos al botón aceptar. Lo primero que vamos a hacer aquí es preguntar qué vale la variable "Opcion". Si su valor es "Alta" comprobar si hemos rellenado todos los campos, ya que a mi entender, todos son obligatorios. Para ello haremos una función que devolverá verdadero en caso de que todos los campos estén llenos, o falso si alguno de ellos no lo está. El código sería el siguiente:

```
Function ComprobarCampos() As Boolean
   If TxtIdCliente = "" Or TxtNomCliente = ""
         Or TxtTelCliente = "" Or TxtDirCliente
         = "" Then
         MsgBox "Faltan campos por rellenar"
      ComprobarCampos = False
   Else
      ComprobarCampos = True
   End If
End Function
```

Como podéis ver es sencilla, preguntamos por todos los campos, y si uno de ellos está en blanco, devolvemos falso y mostramos un mensaje de advertencia.

Muy bien, entonces lo primero que debemos hacer en el botón Aceptar es preguntar si es un alta. Posteriormente, debemos mirar que todos los campos estén llenos, y para acabar, deberíamos comprobar que el IdCliente que hemos introducido en la caja de texto no está repetido, ya que es campo clave y estaríamos provocando un error si intentáramos dar de alta un IdCliente repetido.

Esto es fácil, tan solo tenemos que posicionar el "Recordset" al principio y efectuar un "Find" por el IdCliente, para ver si existe. Inmediatamente después de la búsqueda, preguntaremos si el "Recordset" esta en EOF, porque de ser así, estaríamos a final de fichero, por lo que no habrá encontrado ningún registro con ese IdCliente, así que será correcto. Prestad mucha atención a lo que viene ahora.

Una vez pasada todas las comprobaciones, vamos a añadir el registro. La propiedad que indica a un "Recordset" que se le va a añadir un nuevo registro es ".AddNew", pasándole posteriormente todos los valores (RsRecordset("IdCLiente")=TxtIdCliente, RsRecordset("NomCliente")=TxtNomVliente...) y llamando a la propiedad ".Update" finalmente para finalizar el proceso de alta. Por ahora el botón Nuevo quedaría así:

```
Private Sub CmdAceptar_Click()
If Opcion = "ALTA" Then
   If ComprobarCampos Then
      RsRecordset.MoveFirst
```

```
        RsRecordset.Find "IdCliente=" &
        TxtIdCliente
    If RsRecordset.EOF Then
        RsRecordset.AddNew
        RsRecordset("IdCliente") = TxtIdCliente
        RsRecordset("NomCliente") = TxtNomCliente
        RsRecordset("TelCliente") = TxtTelCliente
        RsRecordset("DirCliente") = TxtDirCliente
        RsRecordset.Update
        LlenarList
        Habilitar
        Frame1.Enabled = False
    Else
            MsgBox "El Id ya existe", vbCritical
    End If
  End If
End If
End Sub
```

Vemos que justo después del "Update" llenamos de nuevo el "ListView", habilitamos los botones Nuevo, Modificar, Borrar y deshabilitamos Aceptar y Cancelar, pero solamente en el caso de que la operación se realice con éxito, porque en caso de que falle algo, no debemos hacer nada, sino mostrar un mensaje que ayude al usuario a corregir su error.

El botón Modificar es prácticamente igual al botón Nuevo, con la diferencia que no debemos limpiar los campos, ya que estamos modificando, y que debemos deshabilitar el campo TxtIdCliente, porque al ser clave, no debe de ser modificado.

```
Private Sub CmdModificar_Click()
   Frame1.Enabled = True
   TxtIdCliente.SetFocus
   Opcion = "EDITAR"
   Deshabilitar
   TxtIdCliente.Enabled = False
End Sub
```

Bien, ahora debemos codificar de nuevo el código del botón Aceptar. Justo después de cerrar la última sentencia condicional, la que preguntaba por el valor de "Opcion", abrimos otra sentencia condicional, que pregunte también por el valor de "Opcion" y si es "EDITAR", que entre dentro. Para modificar un registro debemos posicionarnos primero en él y después modificarlo. Os voy a poner el código necesario para ello, comentándolo posteriormente, pero preferiría que vosotros intentarais entenderlo antes de leer la explicación, así que, vamos, echadle ganas:

```
If Opcion = "EDITAR" Then
   RsRecordset.MoveFirst
        RsRecordset.Find "IdCliente=" &
        TxtIdCliente.Text
   If ComprobarCampos = True Then
        RsRecordset("IdCliente") = TxtIdCliente
        RsRecordset("NomCliente") = TxtNomCliente
        RsRecordset("TelCliente") = TxtTelCliente
        RsRecordset("DirCliente") = TxtDirCliente
        RsRecordset.Update
        LlenarList
        Habilitar
        Frame1.Enabled = False
        TxtIdCliente.Enabled = True
   End If
End If
```

¿No era tan difícil, verdad? Y es que según se mire, se parece bastante al alta. Lo primero que hacemos es posicionarnos al principio y buscar el registro, ya que así nos colocamos en él. Para buscarlo usamos el ya conocido ".Find", y como criterio, el valor que hay en la caja de texto TxtIdCliente.

Una vez buscado, comprobamos los campos, ya que es posible que a alguien se le ocurra borrar el texto de alguno de los "TextBox", por lo que hay que volver a revisarlo. En caso de que todo esté bien, realizamos la modificación, ¿cómo?, pues primero pasándole todos los valores y posteriormente llamando al ".Update". Es muy parecido al alta, con la diferencia de que no hacemos ".AddNew", por lo que el "Recordset" sabe que tiene que modificar y no

añadir.

Seguidamente llenamos el List, para que los cambios surjan efecto, habilitamos y deshabilitamos lo necesario.
Ya solo nos queda lo más sencillo, el borrado. Codificamos el botón borrar como hemos hecho con todos, el código sería el siguiente

```
Private Sub CmdBorrar_Click()
    Frame1.Enabled = True
    TxtIdCliente.SetFocus
    Opcion = "BAJA"
    Deshabilitar
End Sub
```

Y pasamos al botón aceptar.
Este es el mas sencillo de los tres posibles casos. Tan solo debemos posicionarnos en el registro que queremos borrar y llamar a la propiedad ".Delete"
Sería así

```
If Opcion = "BAJA" Then
    RsRecordset.MoveFirst
        RsRecordset.Find "IdCliente=" &
        TxtIdCliente.Text
    RsRecordset.Delete
    LlenarList
    Frame1.Enabled = False
    Habilitar
End If
```

Como veis, nos colocamos al principio, buscamos el registro, lo borramos y listos. ¡Ah!, que no se me olvide, el botón cancelar, que sería como un volver atrás.

```
Private Sub CmdCancelar_Click()
    Frame1.Enabled = False
    Habilitar
    Limpiar
    TxtIdCliente.Enabled = True
End Sub
```

Sencillo ¿no?. Pues con esto hemos acabado por hoy. Espero que no os resulte muy difícil el tema de acceso a datos, pero tened en cuenta que intento introduciros en un par de artículos lo que a mí me explicaron en un año entero de estudios. Ahora tenéis conocimientos para hacer cosas realmente "guapas", es decir, hacer programas útiles y poder decir, "lo he hecho yo". Por ejemplo, os propongo que hagáis una agenda, todo lo completa que vosotros creáis, o retocad el programa actual a vuestro gusto. Así pues, adelante, y suerte!!!

El código completo del programa se encuentra disponible en nuestra WEB (www.hackxcrack.com)en la sección de descargas.

XML: EL FUTURO DE LA TRANSFERENCIA DE DATOS

En un futuro muy próximo habrán dos tipos de personas: las que dominen XML y las que no. ¿A qué grupo quieres pertenecer?

MICROSOFT nos prepara una nueva entrega de Windows (LONGORN) y OFFICE 2003 está a punto de ver la luz. ¿Adivinas qué tecnología aplican? Si, XML

XML es un estándar universal, enfréntate a él cuanto antes !!!

Xml es uno de los lenguajes de los que todo el mundo ha oído hablar, pero que pocos se han puesto a investigar. Xml supone una auténtica revolución en la transferencia e intercambio de datos. Y lo mejor de todo es que es muy sencillo de entender e implementar.

QUÉ ES XML

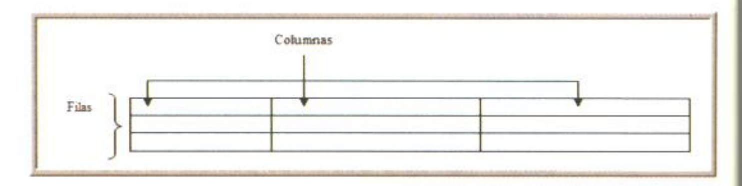
Xml es un lenguaje basado en etiquetas y también un metalenguaje, es decir, un lenguaje que puede ser utilizado para describir otros lenguajes.

Aunque xml y html parece que tengan mucho en común, en realidad, la diferencia entre ambos es igual a la de la noche con el dia . Podemos utilizar html para darle formato a una tabla, pero no podemos utilizarlo para describir los datos que hay dentro de la tabla.

Una tabla es...

Una tabla es un conjunto de filas y columnas, algo parecido a lo que vemos cuando abrimos una hoja de Excel.

Visualmente lo veríamos de la manera siguiente, siendo lo que vemos a continuación lo que podemos "pintar" con html:



Podríamos darle un grosor a las líneas, una separación entre columnas, entre filas, pero no podríamos utilizarlo para describir el contenido.

El contenido lo podemos describir con xml, el contenido lo podemos situar en cada una de las celdas:

Contenido	Contenido	Contenido
Contenido	Contenido	Contenido
Contenido	Contenido	Contenido

Una vez tenemos descrito el contenido , lo podremos formatear con html, es decir darle tamaño a la fuente, colorearlo etc.

Contenido	Contenido	Contenido
Contenido	Contenido	Contenido
Contenido	*Contenido*	**Contenido**

Lo que hace tan poderoso a XML es que cualquier tipo de datos (incluso los datos de concepción abstracta) puede ser estructurado, es decir que **cualquier tipo de datos puede ser construido según una estructura.**

Un tipo de datos de concepción abstracta es

aquel en que determinada información está agrupada según un criterio. Brrrrrrr, vaya frase mas pedante que me ha salido, ejem, veámoslo con un ejemplo: la determinada información que nos interesa de un distribuidor (no nos interesa saberlo todo, no nos interesa saber que le gusta comer magdalenas con salsa de ajo, nos interesa determinada información del distribuidor) la podemos agrupar según el criterio ----> "¿como nos ponemos en contacto con el distribuidor?".

Así pues, tenemos un tipo abstracto de datos al que llamamos distribuidor, y que agrupamos según la estructura "¿como nos ponemos en contacto con el distribuidor?".

Supongamos los conceptos distribuidor y catálogo. A parte del **concepto abstracto** de **distribuidor**, podemos haber especificado la **estructura** que describe la información relacionada con el distribuidor, como nombre, teléfono y dirección. Y en cuanto al **concepto abstracto catálogo**, el número de un producto, su nombre , descripción, precio unitario.... todo ello es posible de especificar con xml.

Ejemplo...

Ejemplo 1: El catálogo

CONCEPTO ABSTRACTO: CATALOGO
|
|
CRITERIO: ¿Qué necesitamos saber para definir inequívocamente cada uno de los Productos (elementos) que contiene el concepto abstracto llamado catálogo?
|
ESTRUCTURA
Número de Producto
Nombre de Producto
Descripción de Producto
Precio unitario del Producto

** Cuando escuchas la palabra catálogo te imaginas exactamente esto, un libreto con un montón de referencias que describen una serie de objetos y posiblemente sus precios. XML te permite que tu crees esos conceptos definiendolos a partir de un criterio y obteniendo finalmente una estructura. Si tu empresa trabaja con, por ejemplo, tipos de fluidos, en lugar de Nombre de Producto podríamos tener Densidad Relativa y en Lugar de Descripción del Producto podríamos tener Porcentaje de Carbono. Eres Tú quien define los conceptos abstractos!!!

Ejemplo 2: Distribuidor

CONCEPTO ABSTRACTO: DISTRIBUIDOR
|
|
CRITERIO: ¿Qué necesitamos saber para definir ponernos en contacto con el Distribuidor?
|
ESTRUCTURA
Nombre
Teléfono
Dirección

XML es un lenguaje de marcado de texto (tiene un conjunto de reglas que especifican como definir etiquetas), definido de manera que no está limitado a un determinado lenguaje, vocabulario o utilización en particular. **Sigue leyendo y entenderás todo esto :)**

QUE PODEMOS HACER CON XML, PARA QUE SIRVE

Personalmente, puedo contaros algo que me ocurría a mi. Leía sobre xml en todas partes, devoraba toda la información que encontraba acerca de este lenguaje y finalmente cuando lo entendía y venia un colega y me hacia la maldita pregunta , se me ponía cara de imbécil

y intentaba salir por la tangente como podía. La maldita pregunta era:

"Esto que me cuentas está muy bien, pero ¿Para que sirve el xml? ¿Puedes darme un ejemplo práctico?"

Y aquí empezaban las divagaciones. Os he recopilado unos cuantos ejemplos, para que os animéis en conocer este lenguaje y sigáis leyendo un poco.

Con XML podréis

• Terminar con el problema de la portabilidad de las bases de datos. Podéis utilizar xml a modo de base de datos tranquilamente.

• Acceder a ficheros de texto de manera indexada, de manera tan sencilla como hasta ahora accedíais a los ficheros ini, multiplicando por bastantes enteros las posibilidades que nos ofrecían estos ficheros.

• Intercambio de datos, xml soporta las estructuras mas complicadas.

• Creación de Servicios Web a través de Soap. Soap es una tecnología de mensajería basada en xml. Los Servicios Web la utilizan para ubicarse. Los servicios Web permiten que las aplicaciones compartan información y que además llamen a funciones de otras aplicaciones independientemente de cómo se hayan creado las aplicaciones, sea cuál sea el sistema operativo o la plataforma en que se ejecutan y cuáles
los dispositivos utilizados para obtener acceso a ellas. Aunque los servicios Web xml son independientes entre sí, pueden vincularse y formar un grupo de colaboración para realizar una tarea determinada.

• Transformar el xml en paginas html. Definir nuevos lenguajes.



Microsoft...

Microsoft, dentro de muy poco sacará al mercado una nueva versión de su flamante Office y anunciará a bombo y platillo que ha "inventado" el XML. Desde PC PASO A PASO te aseguramos que XML será una de las lanzas publicitarias de Microsoft debido a su inclusión en Office y, sinceramente, nosotros ya lo hemos "saboreado" y podemos decirte que a partir de su comercialización habrán dos de usuario de Office: los que saben XML y los que no... ¿A qué grupo quieres pertenecer?

Estamos seguros que las empresas tendrán muy en cuenta tus conocimientos en XML

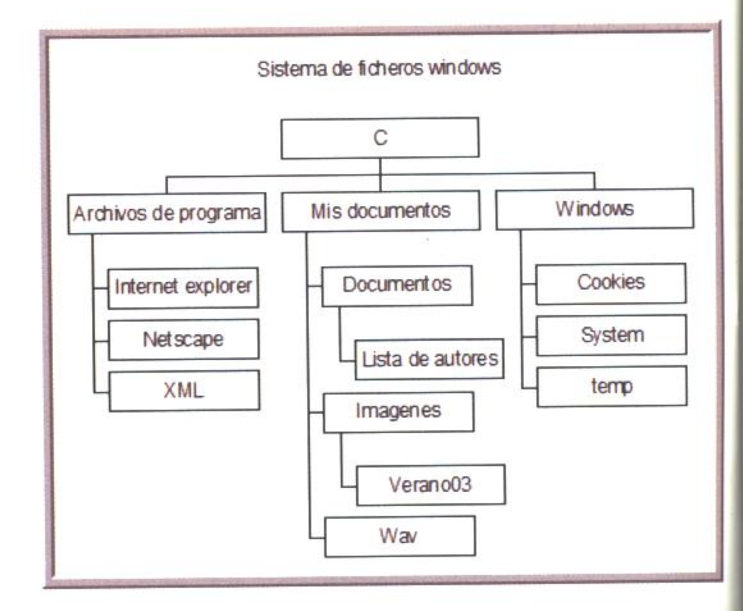
Vamos al grano.

EL LENGUAJE XML

Antes de empezar a codificar en XML deberemos pensar que es lo que queremos estructurar y cómo vamos a estructurarlo.

Un documento o archivo Xml tiene una estructura jerárquica de grafo dirigido. Me explico.

Una estructura jerárquica es aquella en

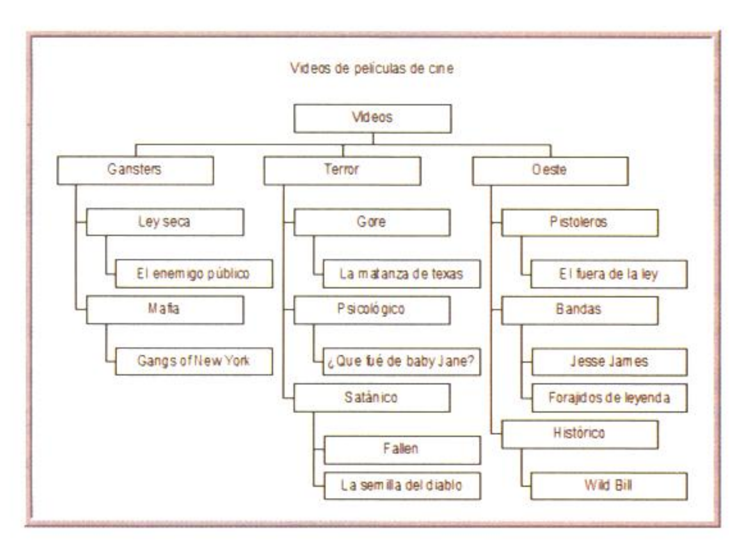


que un objeto esta contenido en otro objeto (a excepción del primero o raíz) y que a su vez es contenedor de otros objetos, y así sucesivamente. Podéis verlo claramente en el explorador de archivos del sistema operativo Windows. Si lo abrís, veréis que todos los directorios cuelgan de un directorio raíz (normalmente se llama C:) y dentro de estos directorios hay otros que a su vez contienen otros.

Lo vemos con eln gráficoanterior.

Pues esta estructura jerárquica la podemos utilizar para organizar cualquier tipo de información.

Supongamos que vamos a organizar nuestros videos (o dvd's):



Esto seria una estructura jerárquica, cada elemento apunta a los elementos que contiene. Videos apunta a Gansters, Terror y Oeste; Satánico a Fallen y a la semilla del diablo. Una estructura jerárquica de grafo dirigido apunta a los elementos que contiene , al elemento que lo contiene (si los hay) y a si mismo.

Por ejemplo Gansters apunta a los elementos que contiene: Ley Seca y Mafia; al elemento que lo contiene : Videos; y a si mismo: Gansters.

Los documentos xml pueden generarse utilizando cualquier editor de texto, como el notepad (Windows) o el vi (Linux). Aunque podéis guardarlo en la extensión que queráis, se recomienda hacerlo con la extensión xml , ya que asegura que cualquier navegador de Internet podrá reconocerlo inmediatamente. Los nombres que asignemos a los tags xml deben seguir una serie de reglas:

• Pueden incluir cualquier caracteres alfanumérico a excepción de los caracteres reservados que son mayor que (> , si tenemos que escribirlo lo sustituiremos por la palabra >), menor que (<, si tenemos que escribirlo lo sustituiremos por la palabra <) y ampersand (&, si tenemos que escribirlo lo sustituiremos por la palabra &).

Ejemplo para aplicar un precio cuya correspondencia sea 5>7:

```
<PRECIOS>
<APLICAR> Cuando correspondencia = 5 &lt;7</APLICAR>
</PRECIOS>
```

• No pueden contener espacios en blanco.
• No pueden empezar por un número, ni un guión (-) ni con signos de puntuación. No obstante, el punto es válido. (<pasta.gansa> es un nombre válido).

• No pueden contener el carácter dos puntos (:).

• No pueden comenzar con el conjunto de letras xml, ya sea en minúsculas o en mayúsculas. (ni xml, ni XML, ni XmL etc.).

• No puede haber un espacio después del carácter de apertura de una etiqueta (<) , sin embargo si que puede haberlo antes del carácter de final (>).

Xml es case-sensitive, es decir que **distingue**

mayúsculas de minúsculas.
En xml no es lo mismo <DEPORTES> que <deportes> o <Deportes>, que son tres etiquetas diferentes.

Los **comentarios en xml** se escriben del siguiente modo:

```
<!-- esto es un comentario xml -->
```

y sirven para incluir anotaciones aclaratorias o personales acerca del documento xml en cuestión.

TAGS —-ETIQUETAS

Todo bloque de información xml debe incluirse dentro de una tag. Un tag es una etiqueta que comienza con el signo menor que, contiene un nombre y finaliza con el signo mayor que. También **se les llama etiquetas.**

```
<ESTO_ES_UN_TAG>
```

este seria pues el tag o etiqueta de inicio.

Todo tag debe tener su final. Sabemos que estamos hablando de el final de un tag, porque además de comenzar con el signo menor que, a continuación nos encontraremos con el signo barra invertida (/)l

```
</ESTO ES_UN TAG>
```

este seria pues el tag o etiqueta de fin.

Puede darse el caso de que un tag sea principio y final, como por ejemplo un intro (un retorno de carro)
Y en ese caso el tag será único y llevara el marcado de que es principio y fin. A estos tags se les llama **elementos vacío** y comienzan con el signo mayor que, a continuación viene el nombre del tag , seguidamente la barra invertida de final(/) y el signo de menor que.

```
<SALTO/>
```

Este seria pues un elemento vacío.

ELEMENTOS

Un elemento de un archivo xml estaría conformado por el tag de inicio, el contenido y el tag de fin, por este orden riguroso. Cualquier otro orden seria erróneo.
En xml todo lo que se abre debe cerrarse.

Ejemplo de elemento:

```
<NOMBRE>Joaquim</NOMBRE>
```

ELEMENTOS RAIZ – ELEMENTOS ANIDADOS

El elemento raíz es el primer elemento del documento, y todos los demás elementos están contenidos dentro de este. Todos los demás documentos están anidados dentro de este.
Todo documento bien formado de xml solo tiene un elemento raíz.
Todos los elementos a excepción del raíz solo tiene un elemento padre. Todo elemento solo cuelga de un elemento.

```
<RAIZ>
        <ELEMENTO> ESTE ELEMENTO PERTENECE AL ELEMENTO RAIZ </ELEMENTO>
        <ELEMENTO> TAMBIEN PERTENECE AL ELEMENTO RAIZ</ELEMENTO>
        <OTRA_INFO> TAMIEN PERTENECE AL ELEMENTO RAIZ</OTRA INFO>
        <OTRA_INFO> EL PADRE DE OTRA INFO Y ELEMENTO ES RAIZ</OTRA_INFO>
        <VARIOS>
          <DEPORTES>
             <FUTBOL>SU PADRE ES DEPORTES</FUTBOL>
             <BALONCESTO>SU PADRE ES DEPORTES</BALONCESTO>
         <COMENT>DEPORTES, VARIOS PERTENECEN A RAIZ</COMENT>
            <COMENT2>DEPORTES + SUS HIJOS PERTENECEN A VARIOS</COMMENT2>
          </DEPORTES>
         </VARIOS>

</RAIZ>
```

Os podéis preguntar , y si quiero especificar contenido propio para el elemento RAIZ o para el elemento VARIOS, o para el elemento DEPORTES, ¿Como lo hago?.
Muy fácil, Se añade a continuación de la etiqueta de inicio:

```
<RAIZ> Este es el texto del elemento RAIZ
  <ELEMENTO> ESTE ELEMENTO PERTENECE AL ELEMENTO RAIZ </ELEMENTO>
  <ELEMENTO> TAMBIEN PERTENECE AL ELEMENTO RAIZ</ELEMENTO>
  <OTRA_INFO> TAMIEN PERTENECE AL ELEMENTO RAIZ</OTRA_INFO>
  <OTRA_INFO> EL PADRE DE OTRA_INFO Y ELEMENTO ES RAIZ</OTRA_INFO>
  <VARIOS> Este es el texto del elemento VARIOS
    <DEPORTES> Este es el texto del elemento DEPORTES
      <FUTBOL>SU PADRE ES DEPORTES</FUTBOL>
      <BALONCESTO>SU PADRE ES DEPORTES</BALONCESTO>
      <COMENT>DEPORTES, VARIOS PERTENECEN A RAIZ</COMENT>
      <COMENT2>DEPORTES + SUS HIJOS PERTENECEN A VARIOS</COMMENT>
    </DEPORTES>
  </VARIOS>
</RAIZ>
```

ANIDAR ELEMENTOS CORRECTAMENTE

• La etiqueta de fin de un elemento, no puede escribirse hasta que no se hayan escrito todas las etiquetas de fin de todos los elementos, cuya etiqueta de inicio haya sido escrita después de la etiqueta de inicio de dicho elemento. Pufffffff, veámoslo con un ejemplo.

Supongamos la etiqueta de inicio "VARIOS":

```
<VARIOS>
  <DEPORTES>
    <FUTBOL>SU PADRE ES DEPORTE</FUTBOL>
    <BALONCESTO>SU PADRE ES DEPORTES</BALONCESTO>
    <COMENT> DEPORTES, VARIOS PERTENECEN A RAIZ</COMENT>
    <COMENT2>DEPORTES + SUS HIJOS PERTENECEN A VARIOS</COMMENT2>
  </DEPORTES>
</VARIOS>
```

no podremos escribir la etiqueta de fin </VARIOS> hasta que hayamos cerrado "DEPORTES", "FUTBOL" ,"BALONCESTO", "COMENT" Y "COMENT2".

no podremos escribir la etiqueta de fin </DEPORTES> hasta que hayamos cerrado "FUTBOL" ,"BALONCESTO", "COMENT" Y "COMENT2".

Esto sería incorrecto:

```
<VARIOS>
  <DEPORTES>
    <FUTBOL>SU PADRE ES DEPORTES</FUTBOL>
    <BALONCESTO>SU PADRE ES DEPORTE>
  </DEPORTES>
    <COMENT> DEPORTES, VARIOS PERTENECEN A RAIZ</COMENT>
    <COMENT2>DEPORTES + SUS HIJOS PERTENECEN A VARIOS</COMMENT2>
  </DEPORTES>
</VARIOS>
```

y seria incorrecto porque baloncesto está pendiente de Cerrar, sin embargo lo que seria correcto es lo siguiente:

```
<VARIOS>
  <DEPORTES>
    <FUTBOL>SU PADRE ES DEPORTES</FUTBOL>
    <BALONCESTO>SU PADRE ES DEPORTES</BALONCESTO>
  </DEPORTES>
    <COMENT> DEPORTES, VARIOS PERTENECEN A RAIZ</COMENT>
    <COMENT2>DEPORTES + SUS HIJOS PERTENECEN A VARIOS</COMMENT2>
</VARIOS>
```

y seria correcto porque no hay ningún elemento abierto dentro de DEPORTES que no haya sido cerrado antes de cerrar DEPORTES. Por otro lado os podéis dar cuenta de la importancia de indentar (dar márgenes a las líneas de

texto) el texto de manera correcta. Tal como está escrito ahora , de un primer vistazo puede parecer que COMMENT y COMMENT2 son hijos de DEPORTES, cuando ahora estamos diciéndole que son hijos de VARIOS. Este texto , correctamente indentado quedaría como sigue:

```
<VARIOS>
  <DEPORTES>
    <FUTBOL>SU PADRE ES DEPORTES</FUTBOL>
    <BALONCESTO>SU PADRE ES DEPORTES</BALONCESTO>
  </DEPORTES>
    <COMENT> DEPORTES, VARIOS PERTENECEN RAIZ</COMENT>
    <COMENT2>DEPORTES + SUS HIJOS PERTENECEN A VARIOS</COMMENT2>
</VARIOS>
```

b- Para todos lo elementos cuyo contenido sea diferente a vacío, debe incluirse una etiqueta de inicio y una de fin.

No es correcto escribir:

```
<DEPORTES>
  <FUTBOL>SU PADRE ES DEPORTES SU PADRE ES DEPORTES</BALONCESTO>
</DEPORTES>
```

a FUTBOL le faltaría la etiqueta de fin y a BALONCESTO la de inicio.

c- A excepción del elemento vacío, no puede escribirse una etiqueta de cierre sin antes haberse escrito una etiqueta de inicio.

El orden debe ser estricto no puede escribirse:

```
</FUTBOL>SU PADRE ES DEPORTES <FUTBOL>
```

Otra muestra de xml mal formado seria:

```
<DEPORTES>
  <FUTBOL><BALONCESTO>SU PADRE ES DEPORTES
  </FUTBOL> </BALONCESTO
</DEPORTES>
```

Fijaros que FUTBOL contiene a BALONCESTO y FUTBOL se cierra antes de cerrar BALONCESTO.

DOCUMENTOS BIEN FORMADOS

El conjunto de todos los elementos de un archivo xml se denomina documento.
Los documentos deben estar bien formados, y estarán bien formados siempre que cumplan los siguientes requisitos:

- Tendrán como mínimo un elemento.
- Todos los elementos estarán comprendidos dentro del elemento raíz.
- Los elementos estarán anidados correctamente.
- No tiene necesidad de validarse contra ningún DTD, ni contra ningún Xml-Schema (encontrareis las explicaciones de estos dos términos en el apartado "LA DECLARACION XML).

LA DECLARACIÓN <?XML>

La declaración <?XML> es la forma por la que el software identifica un documento xml.

Comienza con los caracteres "<?xml" y finaliza con los caracteres "?>".

Versión es obligatorio, pero encoding y standalone son opcionales.

Los atributos version, encoding y standalone deben aparecer en este orden.

La declaración debe estar al comienzo del archivo xml.

```
<?xml version "1.0" standalone="yes" ?>
```

Con esta declaración le estamos diciendo a los programas que interpreten el documento, que se trata de un documento xml, y que está codificado según la version 1.0.

El ultimo dato que incluimos en la declaración es el de que efectivamente, que es un documento standalone. Esto significa que es un documento autosuficiente, que no necesita de ninguna estructura ajena para validarse.

Aunque no vamos a hablar en este artículo de DTD ni de Xml-schemas, creo que es necesaria una pequeña definición para entender el concepto del atributo standalone.

Hemos hablado de que un xml debe estar bien formado, también existe la posibilidad de que el documento xml esté basado en una estructura externa , en una definición de tipo externa o lo que es lo mismo un DTD. Es decir, podemos validar un documento xml contra un archivo DTD.

Imaginaros que queréis generar una serie de documentos xml que queréis que tengan la misma estructura, y que contengan el mismo tipo de datos. Os aseguráis de ello con un DTD. O con un Schema –Xml que viene a ser lo mismo. (DTD = o muy parecido a un XML-Schema).

En el caso de que un documento Xml se valide contra un DTD o contra un Xml-Schema, entonces le diremos a la declaración xml que standalone= "no", o lo que es lo mismo, que el documento xml no es autosuficiente, que necesita de un documento externo para validarse (para ver que sigue las reglas establecidas en el DTD).

Se dice que un documento xml es válido si cumple las restricciones indicadas por su DTD o por su xml-schema, en el caso de que las tuviese, o sea que fuera standalone = "no".

Podemos observar que un documento válido será siempre un documento bien formado, sin embargo un documento bien formado no tiene por que ser válido (es muy posible que sea standalone="yes" o sea que no necesite ni DTD ni xml-schema).

Hemos hablado también del atributo encoding, que no hemos especificado. El atributo encoding sirve para indicar la codificación de caracteres utilizada en el documento (iso, unicode aceptado). Lo haremos solo en casos especiales, cuando tengamos que codificar en japonés, griego, ruso, coreano etc. ,ya que de manera transparente (no tenemos que hacer nada) xml ya utiliza dos.

La declaración para hacer un documento xml en japonés bajo Unix sería:

```
<?xml version "1.0" encoding="EUC-JP" standalone="yes" ?>
```

ATRIBUTOS

Se suelen utilizar los atributos para describir la estructura de datos que estamos construyendo. Los atributos vendrían a ser los adjetivos, los calificadores de un elemento. Estos atributos están contenidos dentro de una etiqueta de inicio (<etiqueta atributo>).

así por ejemplo , si tenemos la siguiente estructura xml:

```
<PROYECTO>
        <NOMBRE> Proyecto X</NOMBRE>
</PROYECTO>
```

También la podríamos codificar de la siguiente manera:

```
<PROYECTO NOMBRE="Proyecto X"></PROYECTO>
```

Si nos fijamos en el ejemplo , vemos que el atributo va entre comillas, y es que todos los valores que asignemos a un atributo deben ir entre comillas.

Sus características son las siguientes:

• Tienen un nombre (PROYECTO NOMBRE) .

• Ese nombre debe de estar seguido de un signo igual y de un par de comillas (aunque no le asignéis contenido) EJ: <PROYECTO NOMBRE=""> seria correcto.

• El tipo de dato que pueden contener, es el que pueden recibir (un atributo puede recibir un valor un momento dado), o el valor de una lista .

• Se les podrá decir si son de carácter opcional, obligatorio o constante desde el DTD, lo comento solo a título de curiosidad, como he explicado, no vamos a entrar en los DTD.

CUANDO UTILIZAR ATRIBUTOS EN LUGAR DE ELEMENTOS HIJOS

Puesto que pueden utilizarse tanto los atributos como los elementos hijos, tal como hemos visto en el ejemplo del proyecto x , uno se puede preguntar cuando utilizar uno o cuando utilizar el otro.

Pregunta de difícil respuesta.

Desgraciadamente no hay una respuesta clara al respecto. Oficialmente, los atributos son pares de nombre-valor, es decir nombre atributo – valor atributo (nombre="proyecto x" en el ejemplo).

Sin embargo hay otras personas que argumentan que la información contenida en los atributos son metadatos .
Los metadatos intentan responder a las preguntas quién, que, cuando, donde, porqué y cómo, sobre cada una de las facetas relativas a los datos que se documentan. Por ejemplo supongamos el número de serie de un CD de música, la mayoría de aplicaciones no necesitarían esa información, entonces puede tener sentido que ese dato sea un atributo. Esto separaría los datos que precisan la mayoría de las aplicaciones , de los datos que la mayoría de aplicaciones no necesitan, y ahí estaría nuestro atributo. Es una posibilidad. Un elemento ocupa mas espacio que un atributo, ¿porque no utilizar entonces un atributo? Primero de todo y mas importante, porque se perdería una de las características básicas de xml, que es la sencillez, la capacidad de autodescripción de las etiquetas y la legibilidad del código xml. También tenemos que considerar que trabajar con elementos, nos dará mas juego para futuras ampliaciones del código, del árbol xml. En cuanto al tamaño, con una buena técnica de compresión nos vendríamos a quedar a la par que las etiquetas con atributos.

Os pongo también una serie de criterios bastante aclaradores:

• Si el dato contiene subestructuras= elemento.
• Si el dato es de gran tamaño = elemento.
• Si el dato cambia frecuentemente = elemento.
• Si el dato es de pequeño tamaño y raramente cambia = atributo.
• Si el dato solo puede tener unos cuantos valores fijos = atributo.
• Si el dato se va a mostrar a un usuario o aplicación = elemento.
• Si el dato no se va a mostrar = atributo.

Utilizad pues un atributo cuando sea un par atributo-valor (algo indivisible, que cuando mencionéis el atributo penséis en el valor, o algo que este muy ligado al atributo), cuando sea un metadato (cuando sea un dato que exista pero que no lo va a consultar todo el mundo, solo casos especiales) y sobretodo cuando os sintáis mas cómodos trabajando con atributos que con elementos.

SECCIONES CDATA

Cuando nos encontremos que tenemos que escribir muchos signos > (>) o < (<) o & (&am;) nuestro código puede convertirse en un auténtico laberinto de símbolos y puede

hacerse totalmente ilegible. Para estos casos es preferible utilizar la sección CDATA. Cuando usamos esta sección, le estamos diciendo al analizador del xml que no analice nada que se encuentre dentro de esta sección, que se la salte, vaya.

Escribiremos el signo mayor que , seguido de un signo de exclamación final, abriremos un signo de grupo [seguido de la palabra CDATA , otro signo de grupo, le añadimos el texto en cuestión, cerraremos dos signos de grupo y escribiremos el signo menor que.

```
<![CDATA [ Texto que queremos escribir]]>
```

Ejemplo:

```
<COMPARAR> <![CDATA [ Si y solo si el num. devuelto es > que 7 y 7 > que el < numero primo] ]>
```

MANIPULAR DOCUMENTOS XML: EL DOM

Ahora podéis estar pensando : que fácil es de crear un archivo xml, pero, ¿como accedo a sus elementos?, ¿como puedo navegar por el? . **La respuesta es el DOM**, un modelo de objetos (un esqueleto, una referencia a los posibles objetos que puede contener un documento xml, objetos tales como comentarios, atributos, elementos....) , con el que no solo podremos navegar, sino también cambiar contenidos de elementos, eliminar y crear nuevos elementos y también copiarlos.

El DOM es independiente de cualquier lenguaje de programación, aunque los mas populares para manipularlo han sido hasta ahora javascript y visual basic y en tercer lugar C.

Este modelo de objetos nos servirá para procesar documentos xml desde programas escritos en Visual Basic, VBA (Visual Basic for aplications, el visual basic que viene adjunto a ms word, ms access, ms Excel etc.), VBScript, C, JavaScript....

Una de las implementaciones del DOM mas populares (la que el 90 % de vosotros tenéis en vuestros ordenadores) es la de Microsoft , y que se instala (para variar) de manera silenciosa en vuestras casas mientras instaláis el Internet Explorer , a partir de la versión 5.

Se distribuye de manera gratuita, así que si queréis conseguir la ultima version de MSXML (si , el DOM de Microsoft) podéis hacerlo desde:

http://www.microsoft.com/xml

No vamos a extendernos en hablar del DOM, lo dejamos para un artículo posterior, simplemente que sepáis que toda esta información que estáis estructurando en un documento xml, la podréis manipular más adelante con este modelo de objetos.

Sin embargo....
Esta bien, para los impacientes vamos a construir un rápido ejemplo, que explicaré en el artículo dedicado al DOM. No voy a comentar lo que hacen las líneas de código, ni las decisiones que me han llevado a estructurar el programa de esta manera. Simplemente lo incluyo para que veáis que toda la información que habéis recopilado en el Xml se puede recuperar fácilmente.

Podéis seguirlo paso a paso y construir una aplicación que os lanzará un mensaje con el texto:

"Hola Mundo XML!"
Se supone que tenéis el Visual Basic 6 a ser posible con el Service Pack 3 o superior instalado.

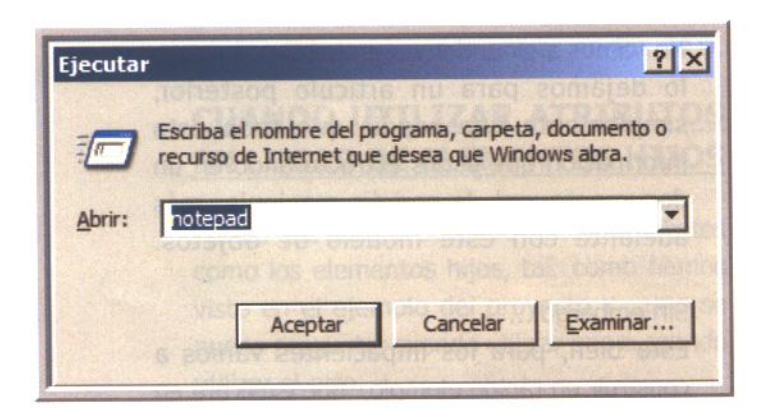
Se supone también un Sistema Operativo Windows, para los desarrolladores Unix mostraremos en un posterior artículo dedicado al DOM, el acceso desde javascript.



Ya explicamos...

Ya explicamos en anteriores número cómo conseguir el Visual Basic, instalarlo y todas esas cosas… bueno venga, una pista para conseguirlo: www.spanishare.com (sin comentarios) ;p

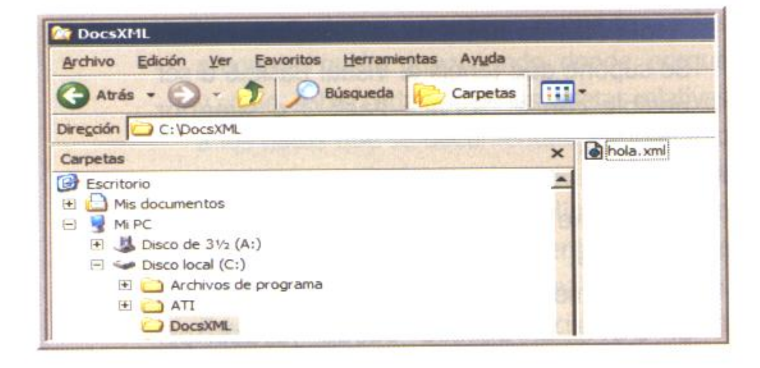
1 – Abrid el **bloc de notas** . Si no sabéis donde lo tenéis porque utilizáis herramientas muy sofisticadas podéis ir a **INICIO/EJECUTAR** Y en la ventanita que os sale teclear: **notepad**.



2- Una vez tengáis el programa notepad abierto, teclear dentro lo siguiente:

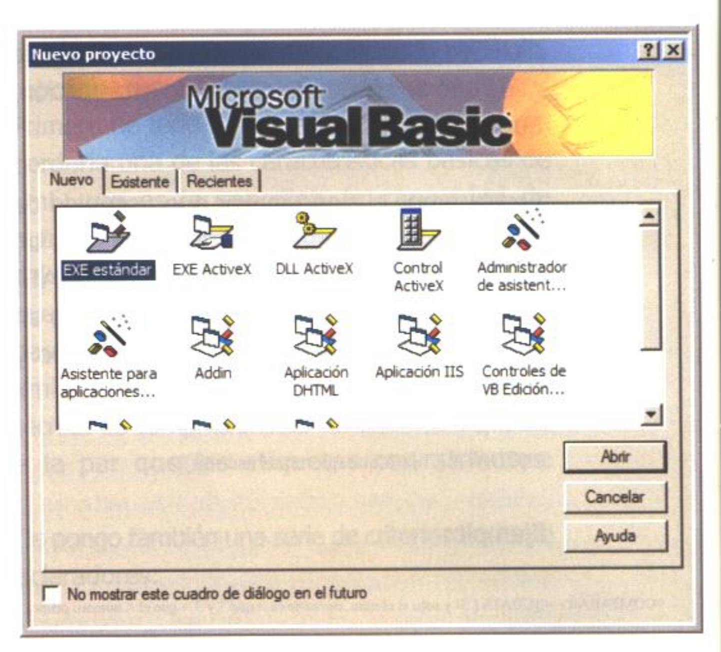
```
<?xml version="1" standalone="yes" ?>
<BIENVENIDA>
<HOLA>Hola Mundo Xml</HOLA>
</BIENVENIDA>
```

3-Crear una carpeta nueva en el explorador de Windows, que se llame docsXML y que cuelgue de C.

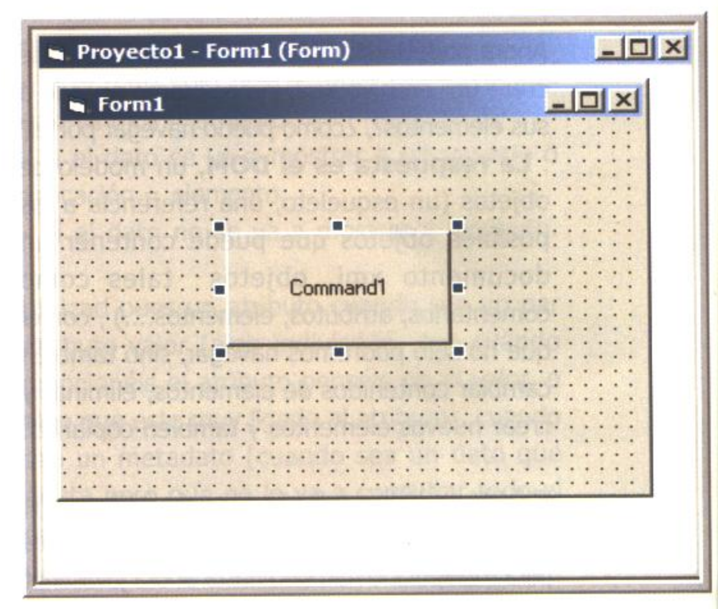


4- Abrid el Visual Basic.

5- Seleccionad un proyecto standard exe.

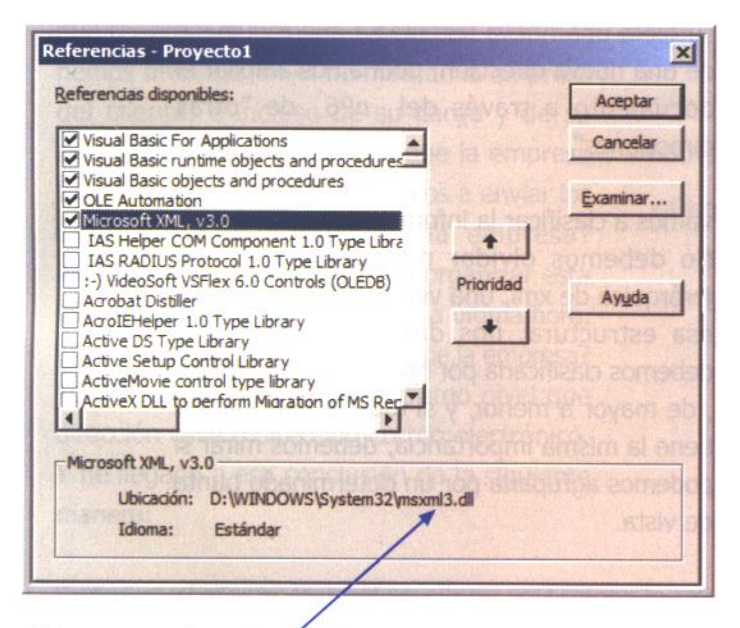


6- En el formulario que os ha abierto añadid un botón en el medio.



7- Añadid una referencia al DOM. Para ello seleccionar del menú PROYECTO la opción REFERENCIAS.

Buscad una que pone Microsoft XML, v3.0 y seleccionarla, haciendo click en el cuadradito que hay a la izquierda de modo que quede marcado. Y dadle al botón de Aceptar



Fijaros que el nombre de la referencia es a la dll msxml3.

8- Haced doble click en el botón del formulario y escribid lo siguiente:

```
Private Sub Command1_Click()

'*******Declaraciones****************************

Dim objXMLDOM As New MSXML2.DOMDocument30

Dim objNodes As IXMLDOMNodeList

Dim path As String

Dim iIndex As Integer

Dim elementos() As Variant

'**************Asignaciones**********************
path = "C:\docsXML\hola.xml"

objXMLDOM.async = False

'compruebo que he puesto bien el path
'MsgBox Dir("C:\docsXML\hola.xml")

objXMLDOM.Load (path)

Set objNodes = objXMLDOM.selectNodes("BIENVENIDA/HOLA")

ReDim elementos(objNodes.length)

'**************Cuerpo********************

For iIndex = 0 To objNodes.length - 1

  elementos(iIndex) = objNodes.Item(iIndex).nodeTypedValue

  MsgBox elementos(iIndex)

Next

End Sub
```

9- Pulsad la tecla F5 y....

¡¡¡¡¡EUREKA!!!! Os sale un mensaje que os dice "Hola Mundo XML", ¿No es bonito esto? Acabáis de acceder al elemento HOLA de vuestro documento xml y habéis visualizado su contenido.
Ahora igual os habéis quedado mas tranquilos, o igual os pasa como me ocurrió a mi cuando ví aparecer en pantalla el primer , reluciente, fulgurante, rutilante literal y quería mas, mas, mas.....jajaja

Uf que subidón.

CREACIÓN DE UN ARCHIVO XML PASO A PASO

1- Entender bien lo que tenemos que hacer o lo que queremos hacer

Es muy importante no empezar a trabajar "sobre la marcha", saber exactamente lo que vamos a hacer hasta sus mínimos detalles. A veces un detalle aparentemente absurdo ha supuesto un esfuerzo tremendo de adaptación a nuestro

trabajo.
Vamos a hacer algo muy sencillo , tenemos que clasificar la información de todas las direcciones que puede tener un cliente potencial.

2- Clasificación de la información

Una vez entendemos bien lo que debemos hacer, vamos a plasmar nuestros pensamientos, sobre papel, en un archivo de texto de nuestro PC... donde queramos. Tal como nos venga, y una vez lo tengamos todo plasmado, lo ordenaremos clasificándolo.

Supongamos ese cliente potencial, a la voz de pronto me vienen a la cabeza...

1. Dirección comercial.
2. Dirección personal.
3. Dirección de correo electrónico que tiene en la empresa.
4. Direcciones de correo electrónico particulares.
5. Direcciones de sus paginas Web si las tuviese.
6. Otras direcciones.
7. Necesito saber a nivel de direcciones de calles: el numero, el nombre de la calle, la planta , la escalera, la ciudad, la provincia, el numero de código postal
8. Es todo?

Bueno ahora nos quedaría pulir esto un poco, vamos a empezar por "otras direcciones". ¿Cuales pueden ser estas "otras direcciones"? ¿Las direcciones de los bancos con que trabaja?,no , si seguimos ese criterio acabaremos colocando en la lista de "otras direcciones", las direcciones de las discotecas donde bebe cerveza negra. Vamos a ceñirnos a las direcciones a las que podemos dirigirnos y por lo que veo estas serian finalmente todas las direcciones. Borramos el numero 8 ("es todo?") pero sin embargo dejamos el nº 6 de "otras direcciones". En el caso de que se nos ocurrieran nuevos tipos de direcciones o surgiera una nueva tecnología que requiriese de una nueva dirección, podríamos ampliar el documento a través del nº6 de "otras direcciones".

Vamos a clasificar la información:
No debemos olvidar nunca la estructura jerárquica de xml, una vez tenemos en mente esa estructura, nos daremos cuenta que debemos clasificarla por niveles de importancia , de mayor a menor, y si toda la información tiene la misma importancia, debemos mirar si podemos agruparla por un determinado punto de vista.

Por ejemplo: Hoy en día es tan importante la dirección de correo electrónico como la personal de la calle. Sin embargo en la dirección personal es donde supuestamente leerá el correo electrónico particular, así como leerá el correo electrónico de la empresa en la dirección de la empresa.
Lo mismo podríamos suponer de las paginas Web, tendrá unas para la empresa y otras a nivel particular.

Hemos comentado que necesitábamos saber a nivel de direcciones de calle: el nombre de la calle ,el numero , el piso, código postal etc. Esta información podría pertenecer a una misma entidad.

Vemos que el numero es de una calle, y esta asociado indivisiblemente al piso y escalera. Asimismo vemos que el código postal es de esa dirección . Podemos pues, crear un elemento dirección calle que este al mismo nivel que dirección url, o dirección correo electrónico. Lo tenemos todo.

Antes de ponernos a estructurar hemos de preguntarnos:

¿Nos olvidamos algo?

Si pensamos detenidamente, vemos que sí ,

que tenemos el nombre del cliente, pero nos hemos olvidado de el nombre de la empresa del cliente, e incluso de su cargo y de su departamento en el caso de que la empresa fuera muy grande. ¿Como íbamos a enviar los correos sin el nombre de la empresa? De acuerdo , ahora nos queda organizar esta información que nos ha surgido a última hora. ¿A que nivel situamos el nombre de la empresa? Yo he propuesto hacerlo al mismo nivel que dirección calle o dirección correo electrónico. Y he llegado a esa conclusión de la siguiente manera:

Supongamos que la empresa crece, que adquiere otra sucursal, ¿Como se llamará la empresa? Tendrá el mismo nombre. Y el cargo del cliente ¿cambiará cuando tengamos una nueva sucursal? No, y tampoco lo hará su departamento. Por ese motivo lo pondré al mismo nivel que las direcciones.

Es importante cuando analicemos un problema pensar en lo que tenemos y en lo que podemos llegar a tener, es importante hacerse las preguntas: ¿Que pasaría si...? , ¿Y si....?.

Bueno pues ya tenemos hecha una posible clasificación. Y digo posible porque cualquier otra, mientras sea coherente y argumentada puede ser tan o mas buena que esta que os he presentado yo.

Ahora vamos a pintarlo en un organigrama para verlo visualmente, para ayudarnos a pintar el xml.

Quedaría así (vease gráfico en la página siguiente).

Para terminar, construiremos el xml con el notepad o el vi o cualquier otro editor de texto, que quedará como sigue:

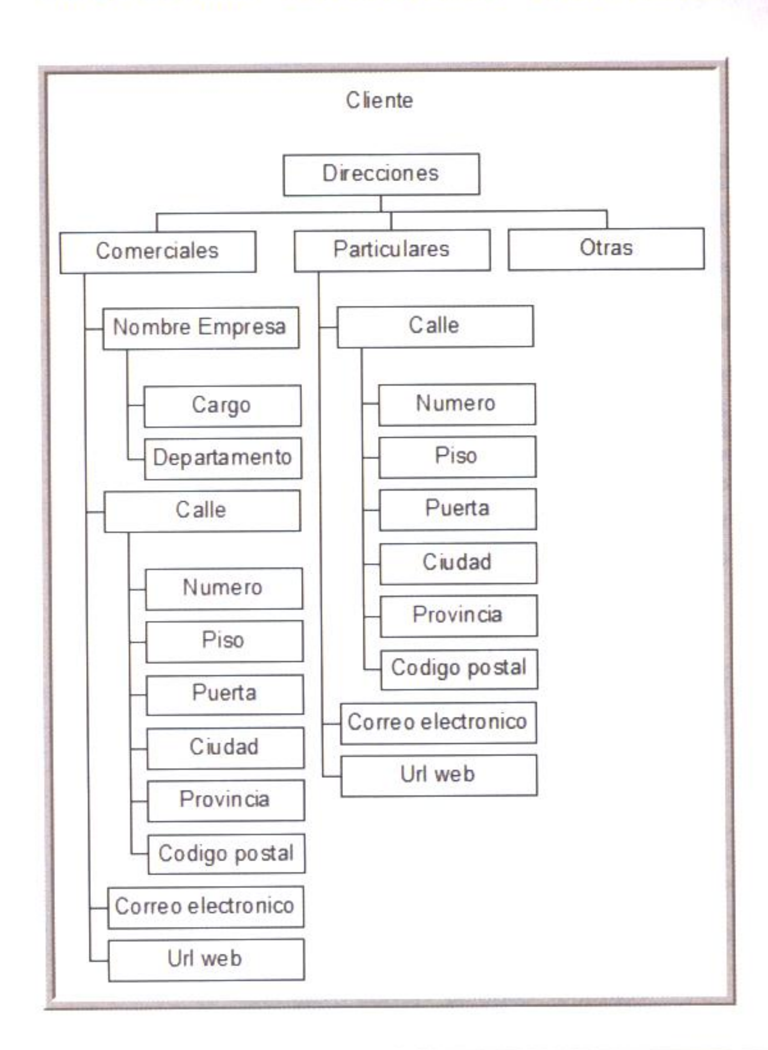
```xml
<?xml version="1.0"  standalone="yes"?>
<!--Tendremos varios clientes, luego cliente
no nos sirve de directorio raiz -->
<Direcciones_Cliente>
    <Cliente>Juan Jose Martinez Ruiz
        <Direcciones>
            <Comerciales>
                <Nombre_Empresa>Azorin
                    <Cargo>Director</Cargo>
                    <Departamento>Desarrollo</Departamento>
                </Nombre_Empresa>
                <Calle>Mallorca 125
                    <Piso>1</Piso>
                    <Puerta>4</Puerta>
                    <Ciudad>Leganes</Ciudad>
                    <Provincia>Madrid</Provincia>
                    <Codigo_Postal>07485</Codigo_Postal>
                </Calle>
                <Correo_Electronico>jj@cor.com</Correo_Electronico>
                <Url_Web>www.generacion98.es</Url_Web>
            </Comerciales>

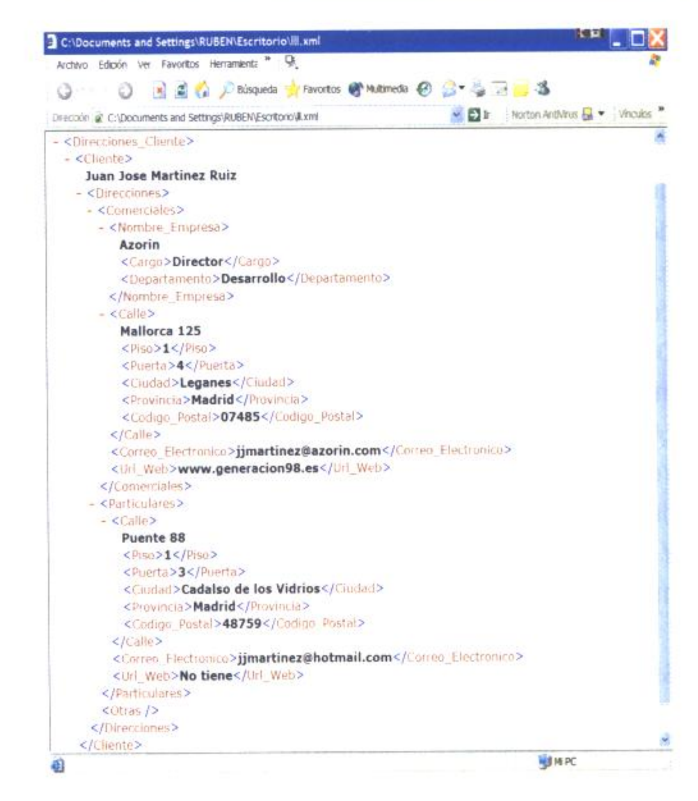
            <Particulares>
                <Calle>Puente 88
                    <Piso>1</Piso>
                    <Puerta>3</Puerta>
                    <Ciudad>Cadalso Vidrios</Ciudad>
                    <Provincia>Madrid</Provincia>
                    <Codigo_Postal>48759</Codigo_Postal>
                </Calle>
                <Correo_Electronico>hxc@jj.com</Correo_Electronico>
                <Url_Web>No tiene</Url_Web>
            </Particulares>
            <Otras>
            </Otras>
        </Direcciones>
    </Cliente>
</Direcciones_Cliente>
```

Lo podéis guardar como direcciones.xml y abrirlo en el Explorer o en netscape para visualizarlo (tienes la imagen en la página siguiente).

Espero que os haya gustado el articulo y que os animéis a programar en xml. Tened en cuenta que solo hemos hablado de una pequeña parte del xml, la parte más básica sin duda, pero solo una pequeña parte.

Saludos compañeros!





Esta es la imagen del archivo XML visto con el Internet Explorer. Por si es tu primera vez, fíjate en los "-" (parecido al símbolo de restar) y pulsa sobre ellos :p

APACHE PARTE IV
TRIO DE ASES
APACHE - PHP - MYSQL

Vamos a montar un FORO como el de www.hackxcrack.com
Instalaremos PHP en APACHE
Instalaremos un Servidor de Bases de Datos
Instalaremos, posiblemente, el mejor FORO que existe

Vamos a divertirnos un poco :)

Hace varios números que estamos liados con APACHE y, para no cansar a la peña con demasiada "teoría" vamos a hacer algo muy, muy, muy práctico y refrescante.

Hemos recibido muchos mails en los que se nos pide que enseñemos cómo montar un Foro al estilo del **Foro de Hack x Crack** (www.hackxcrack.com).

Pues bien, si has seguido nuestro curso, a estas alturas seguro que tienes el Servidor Apache montado en tu ordenador y unos conocimientos sobre el mismo más que suficientes sobre su funcionamiento. Estamos sobradamente preparados para **montar un foro en nuestro equipo y vamos a ponerlo a disposición del mundo :)**

Puesta a Punto y Conceptos Previos

Vamos a montar uno de los foros más completos que existen en la actualidad y para colmo es Software Libre bajo licencia GPL. Resumiendo esto significa que no tendrás que pagar un solo euro por utilizarlo ni aguantar ningún tipo de limitación y por supuesto, ningún tipo de añadido indeseable tipo "spyware-adware"

y/o banners publicitarios.



Licencia GPL...

Licencia GPL: Si te gusta indagar en esto del Software Libre y leerte las licencias puedes hacerlo en http://www.gnu.org/licenses/gpl.html
Spyware: Código que se introduce en algunos programas aparentemente gratuitos pero que se dedican a espiar tus gustos "interneteros" y enviar todo lo que puedan sobre los mismos (y sobre tu identidad) a ves a saber quien para ves a saber qué oscuras intenciones (normalmente para vender tus datos y machacarte a base de spam).
Spam: Resumiendo... correo no deseado.
Adware: Para disfrutar de un programa se te obliga a soportar unos cuantos mensajes publicitarios, normalmente en forma de banners.

Personalmente opino que el spyware es un acto delictivo mientras que el adware es una práctica perfectamente legítima. Lo malo es que en la práctica, la línea que separa ambas técnicas es bastante difusa y casi nunca estás seguro de a qué te estás enfrentando. Si quieres más información sobre el tema puedes pasarte por http://www.persystems.net/sosvirus/general/spyware.htm (esta página contiene algunos enlaces para ampliar tu información y está en perfecto castellano).

phpbb (php bulletin board) es un foro programado en PHP y necesita para su correcto funcionamiento un sistema de base de datos, en este caso utilizaremos MySQL pero admite muchas más. Si eres principiante en esto de los Servidores Web seguro que estás a punto de "pasar" de seguir leyendo... eso de phpbb, PHP y MySQL... buffff, qué complicado ¿no? PUES NO!!! Vas a aprender a instalar los módulos PHP y MySQL en Apache y finalmente instalaremos el foro phpbb y lo pondremos "on line". Lo haremos paso a paso y para colmo lo entenderás perfectamente.

Vamos a meternos de lleno en el **Trío de Ases de los Servidores Web: Apache + PHP + MySQL.** Esta es la configuración más extendida y recomendada actualmente. **Apache** es el **Servidor Web** sobre el que venimos instruyéndote desde hace tres números, **PHP** es un **lenguaje de programación** creado especialmente para trabajar con datos y **MySQL** es la **base de datos** en sí misma. Todo ello será utilizado por nuestro foro :)



Hay algún...

Hay algún paquete de software que te instala todo esto con solo pulsar un icono, pero si estás leyendo esta revista es porque quieres saber lo que hay detrás de las cosas. Desde nuestro punto de vista, los "automatismos" son tremendamente útiles precisamente cuando ya conoces un procedimiento y te ves condenado a repetirlo una y otra vez hasta el hartazgo (imagina que tuvieses que instalar 10.000 foros phpbb) pero darle un proceso automático a alguien que está aprendiendo es negarle el derecho a aprender (y de eso nuestro querido Bill Gates sabe mucho).

OJO!!! Si me meto con nuestro "querido" Tio Bill no es porque nos facilite el trabajo (eso es bueno) sino porque al mismo tiempo nos niega el derecho a poder hacer lo mismo de forma manual. Microsoft intencionadamente elimina y/o no documenta muchos procesos, eso impide a cualquiera aprender (exactamente lo contrario que Linux, que te obliga a hacer muchas cosas de forma manual y por lo tanto te obliga a aprender).

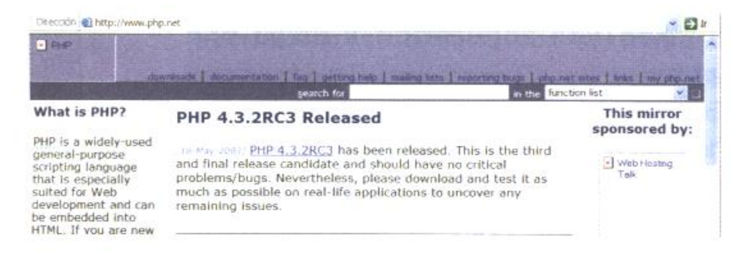
PHP: Hypertext Preprocessor -- Presentación e Instalación:

Partimos de que tenemos instalado nuestro Servidor Web APACHE (explicado extensamente en anteriores entregas, al final de esta revista tienes los índices de antriores números).

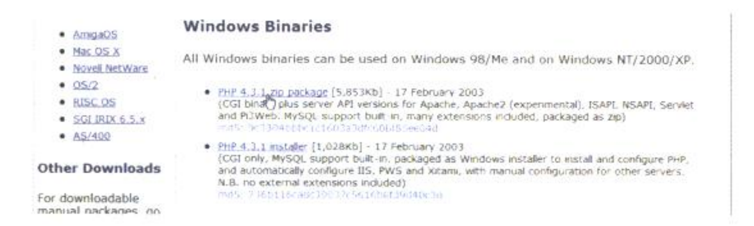
Como ya sabemos APACHE es capaz de "interpretar/servir" páginas con extensión HTM y HTML (entre otras), pero no es capaz de interpretar/servir páginas PHP si no le instalamos el módulo correspondiente. Eso es exactamente

lo que haremos ahora, Instalar el Modulo PHP.

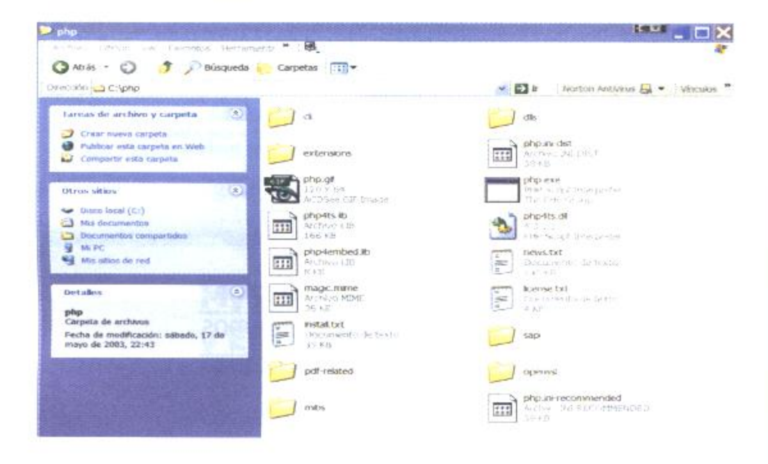
1.- Nos vamos a la página oficial de PHP (www.php.net).



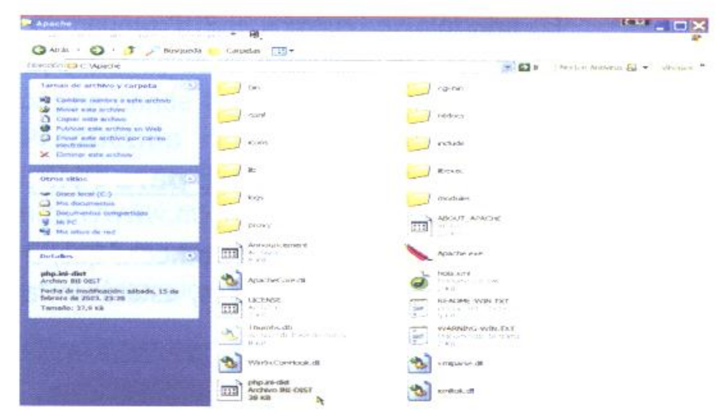
2.- Pulsamos sobre DOWNLOADS (arriba a la izquierda) y nos descargamos el archivo para Windows. En la imagen puedes comprobar que estamos bajando el archivo PHP para Windows **PHP 4.3.1 zip package**, el otro archivo es un instalador pero no lo necesitamos.



3.- Descomprimimos el archivo. Creamos una carpeta llamada php en c:\ y metemos dentro el contenido del ZIP... nos quedará algo como esto:



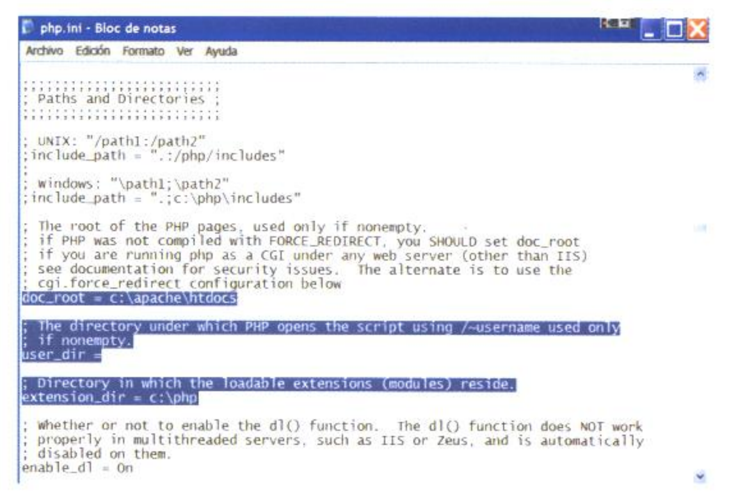
4.- Ahora copiamos el archivo **php.ini-dist** en la carpeta de apache, si seguiste nuestros pasos a la hora de instalar apache lo más seguro es que debas copiarlo en c:\apache.



5.- Una vez copiado en la carpeta de apache cambia el nombre al archivo **php.ini-dist** y déjalo como **php.ini**

6.- Ahora editamos el php.ini con el Block de Notas (o cualquier editor de texto plano). Una vez ante nosotros buscaremos la "palabra" **extension_dir** (en el apartado **Paths and Directories**) y le añadiremos la ruta para que pueda acceder a las librerías de PHP. En nuestro caso tenemos el PHP en c:\php y por tanto quedará **extension_dir = c:\php**

No guardes el archivo, fíjate que un poco más arriba tenemos la "palabra" **doc_root =**, muy bien, pues añádele la ruta de acceso a la raíz Web del servidor apache, en nuestro caso c:\apache\htdocs, quedándonos **doc_root = c:\apache\htdocs**



Finalmente guardamos el archivo.

En principio ya tenemos el PHP preparado, ahora hay que "notificarle" al APACHE que ya puede disponer de PHP :).

Configurando APACHE para PHP:

Vamos a editar el archivo **httpd.conf** que, si has seguido nuestros cursos, estará en la carpeta c:\apache\conf\



Hice seguir...

Hice seguir este texto a un alumno de la ESO y pude ver como le era imposible editar este archivo desde un Windows XP. El problema era que si pulsaba el botón derecho del mouse no le salía la opción de editar con el Block de Notas.

La solución pasa por ejecutar el Block de Notas y abrir el archivo desde el programa (un proceso muy lento), pero yo que utilizo mucho MAC le propuse que abriese el Block de Notas y arrastrase el archivo httpd.conf desde c:\apache\conf\ hasta el Block de Notas. El alumno se sorprendió enormemente de que un archivo pudiese editarse de esa forma y yo me sorprendí de que se sorprendiese tanto (valga la redundancia).

Como soy bastante curioso, estuve proponiendo este tema (el abrir archivos arrastrándolos hacia el programa adecuado) a cuantos "Windows-Maníacos" conocía y muy pocos conocían esta práctica tan útil. Si eres uno de ellos, ya sabes ;)

Buscamos la sección ScriptAlias y añadimos la siguiente línea:

ScriptAlias /php/ "c:/php/"

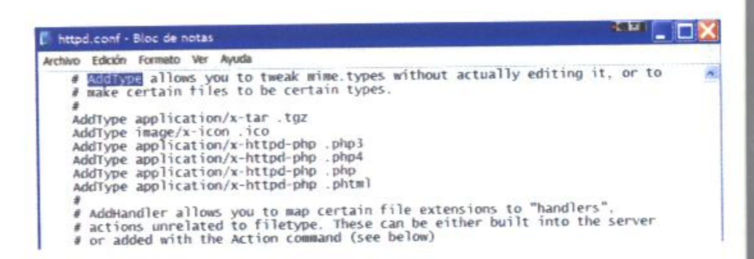
Con esto conseguimos que a partir de este momento el directorio c:\php pueda llamarse php.

```
httpd.conf - Bloc de notas
Archivo  Edición  Formato  Ver  Ayuda
#
# ScriptAlias: This controls which directories contain server scripts.
# ScriptAliases are essentially the same as Aliases, except that
# documents in the realname directory are treated as applications and
# run by the server when requested rather than as documents sent to the client.
# The same rules about trailing "/" apply to ScriptAlias directives as to
# Alias.
#
ScriptAlias /cgi-bin/ "C:/Apache/cgi-bin/"
ScriptAlias /php/ "c:/php/"
#
# "C:/Apache/cgi-bin" should be changed to whatever your ScriptAliased
# CGI directory exists, if you have that configured.
#
```

Buscamos la sección **AddType** y le añadimos las siguientes líneas:

AddType application/x-httpd-php .php3
AddType application/x-httpd-php .php4
AddType application/x-httpd-php .php
AddType application/x-httpd-php .phtml

Con esto conseguimos que cuando APACHE "lea" un archivo con la extensión php3, php4, php o phtml lo reconozca y no nos de error.

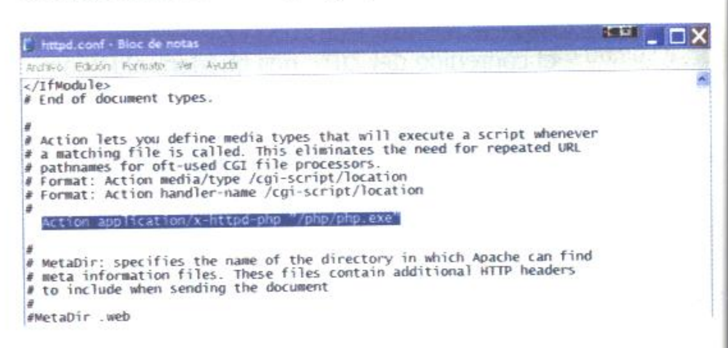


```
httpd.conf - Bloc de notas
Archivo  Edición  Formato  Ver  Ayuda
    # AddType allows you to tweak mime.types without actually editing it, or to
    # make certain files to be certain types.
    #
    AddType application/x-tar .tgz
    AddType image/x-icon .ico
    AddType application/x-httpd-php .php3
    AddType application/x-httpd-php .php4
    AddType application/x-httpd-php .php
    AddType application/x-httpd-php .phtml
    #
    # AddHandler allows you to map certain file extensions to "handlers",
    # actions unrelated to filetype. These can be either built into the server
    # or added with the Action command (see below)
```

Finalmente buscamos la sección **IfModule** y le añadimos la línea:

Action application/x-httpd-php "/php/php.exe"

Con esto le indicamos a APACHE dónde está el ejecutable de PHP. Puedes ver que en lugar de decirle que está en c:\php\php.exe le damos el alias antes definido /php/ :)



```
httpd.conf - Bloc de notas
Archivo  Edición  Formato  Ver  Ayuda
</IfModule>
# End of document types.

#
# Action lets you define media types that will execute a script whenever
# a matching file is called. This eliminates the need for repeated URL
# pathnames for oft-used CGI file processors.
# Format: Action media/type /cgi-script/location
# Format: Action handler-name /cgi-script/location
#
  Action application/x-httpd-php "/php/php.exe"

#
# MetaDir: specifies the name of the directory in which Apache can find
# meta information files. These files contain additional HTTP headers
# to include when sending the document
#
#MetaDir .web
```

¿Funciona PHP?

Para saber si PHP está disponible, crearemos una página PHP que nos haga un diagnóstico de la instalación de PHP.

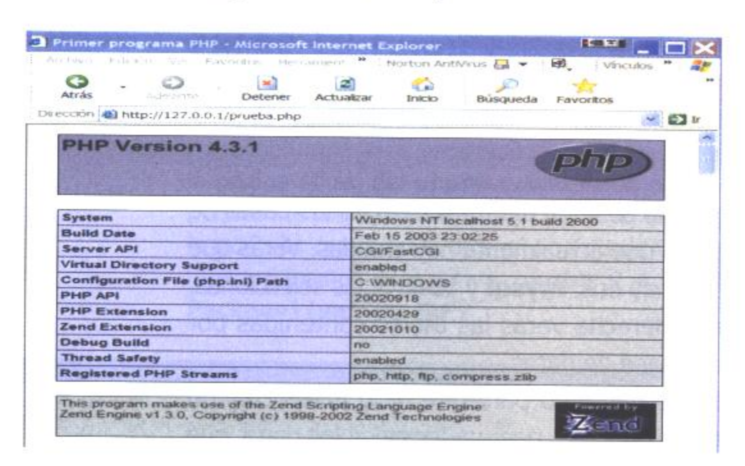
Cogemos el Block de notas y copiamos lo siguiente:

```
<html><head><title>Mi Primera Pagina en PHP</title></head>
<body>
```

```
<?
phpinfo();
?>
</body>
</html>
```

Guardamos el archivo con el nombre prueba.php y lo copiamos en la ruta de acceso a la raíz Web del servidor apache, en nuestro caso c:\apache\htdocs\

Ahora ejecutamos Apache (esto ya lo explicamos en anteriores números), iniciamos nuestro Navegador Web y accedemos a la página http://127.0.0.1/prueba.php Si todo ha ido bien veremos la página de diagnóstico de PHP :)



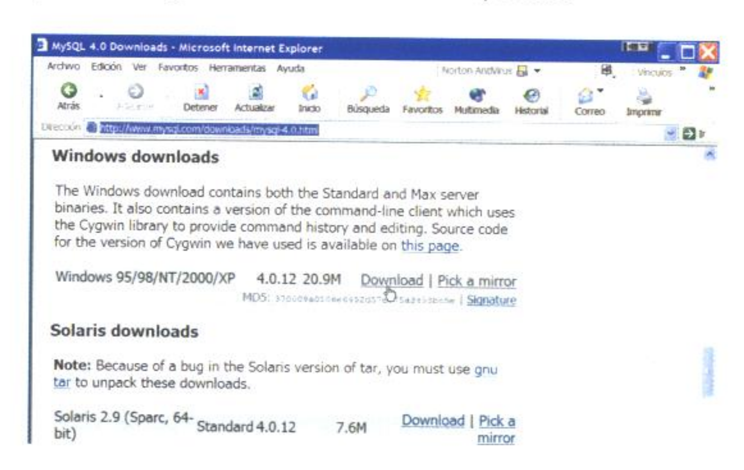
No vamos a...

No vamos a enseñar a programar en PHP en este número de Hack x Crack, lo dejamos para otra ocasión. Pero conste que para seguir este artículo no es necesario saber PHP.

MySQL: Presentación, Instalación y Ejecución

Como ya hemos dicho PHP es el elemento ideal para acceder a una base de datos y MySQL es la base de datos ideal para la ocasión. Pues venga, vamos a instalar MySQL.

Lo primero es visitar la página oficial de MySQL (http://mysql.com) y pulsar sobre la pestaña **downloads** de la Web. Nos aparecerá una página donde buscaremos **MySQL 4.0 -- Production release (recommended)** finalmente seremos conducidos a http://www.mysql.com /downloads/mysql-4.0.html, buscaremos **Windows Downloads** y nos descargaremos el único archivo disponible.



Una vez descargado lo descomprimimos y ejecutamos el setup.exe y, si aceptamos todos las ventanitas que salgan, se nos instalará MySQL en c:\mysql

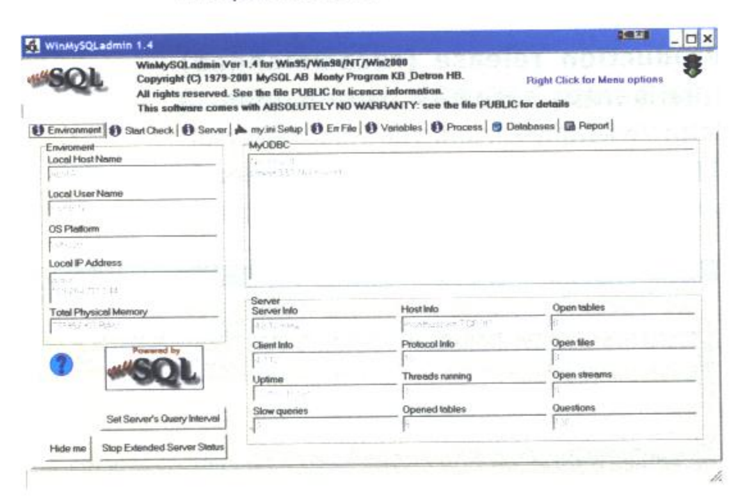
Ahora nos vamos al directorio c:\mysql\bin y ejecutamos el archivo winmysqladmin.exe, nos encontraremos ante la siguiente ventanita



Introduciremos un nombre de **usuario** (en nuestro caso **petor903**) y una **clave** (en nuestro caso **ante9832**). No pongas la misma o cualquier usuario que lea esta revista podrá acceder a tu base de datos desde Internet ;p (no es exactamante así pero haznos caso, pon otro user/pass).

Junto al reloj del sistema te aparecerá un nuevo icono con aspecto de semáforo, te permitirá acceder al Centro de Control. Pulsa el botón

derecho del ratón sobre él y selecciona **Show me** para acceder.



Ante todo tranquilidad, no dejes que te atemorice ¿vale? Ahora miraremos lo que es imprescindible para poder instalar nuestro foro, pero antes déjame comentarte un par de cosas:

- Lo que tienes ante ti es un Servidor de Bases de Datos al que cualquier programa puede conectarse para crear, modificar y/o mantener una Base de Datos. Cuando digo **conectarse** estoy diciendo precisamente eso: **conectarse**.

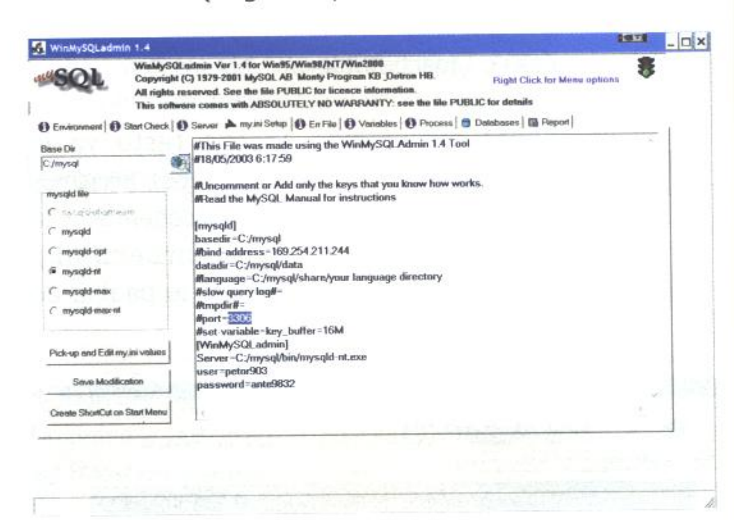
Este Servidor está ahora mismo on-line y escuchando un puerto, está esperando que cualquier cosa (programa) se conecte a él desde Internet o desde la red local.

- Este Servidor de Datos no tiene nada que ver con APACHE, es completamente independiente de cualquier cosa. Podrías tener el Servidor de Datos instalado el Rusia en un MAC y podrías acceder a él perfectamente a través de Internet :)

Venga, vamos a fisgonear un poco :)

En la pestaña **environment** tenemos una serie de datos sobre nuestro PC, pero el más importante es nuestra Local IP Address, que en nuestro caso es 169.254.211.244 (toma nota en un papelito que después la utilizaremos).

En la pestaña **my.ini Setup** tenemos el archivo de inicio de MySQL, es decir, la configuración de MySQL. No nos meteremos ahora en ello, pero toma buena nota del puerto que está escuchando (seguro que es el 3306)



Y en la pestaña databases tienes las bases de datos que actualmente hay creadas. Verás que solo hay dos: **mysql** y **test**. Si las seleccionas, a la derecha verás las tablas contenidas por esa base de datos.

El resto lo dejo para que te lo mires cuando tengas tiempo, pero ya tenemos los datos que necesitamos: IP y Puerto (como siempre que hablamos de Servidores).

Por cierto, si quieres comprueba que se está ejecutando MySQL accediendo mediante Telnet.

1.- Inicia una Ventana de Comandos (Ya explicado mil y una veces en números anteriores).

2.- Escribe telnet IP Puerto, en nuestro caso particular:

telnet 169.254.211.244 3306



Puedes ver la Versión de tu Servidor de Datos (MySQL), en este caso la versión 4.0.12

El Servidor de Datos ha contestado a la petición de conexión por Telnet indicando la versión de MySQL que está corriendo.



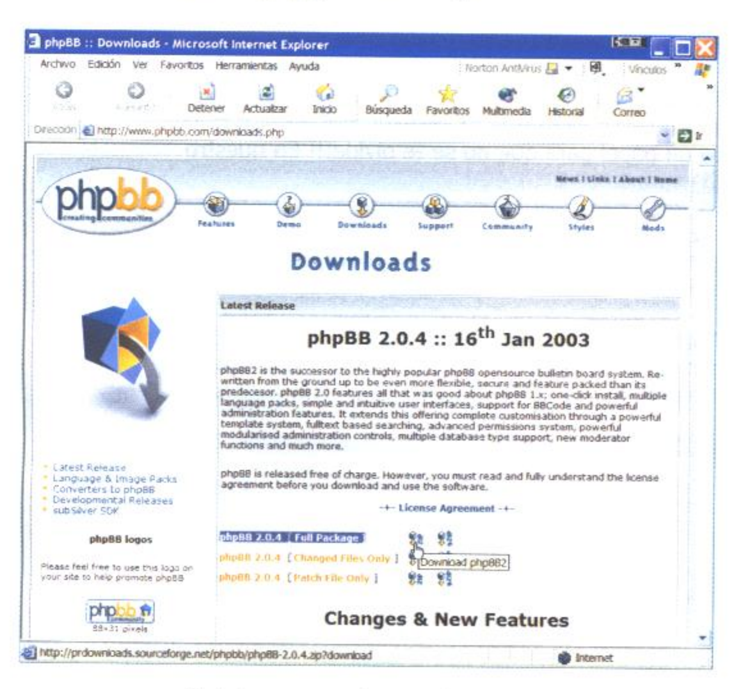
Ya hemos explicado...

Ya hemos explicado en anteriores números sobre el uso de TELNET según la versión de Windows que tengas.

Bueno, ya está. Tenemos PHP instalado, APACHE ya sabe que dispone de PHP y finalmente ya tenemos un Servidor de Datos corriendo en nuestro equipo. Tenemos todo lo necesario para montar nuestro FORO :)

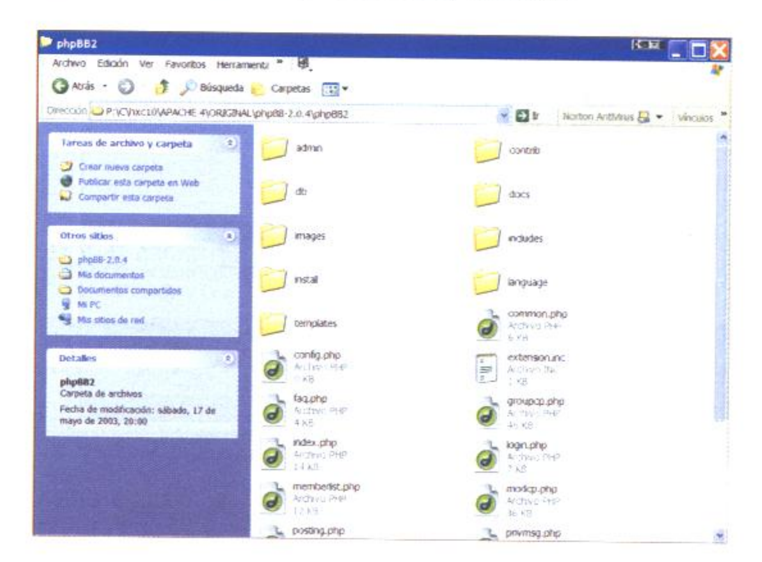
PHPBB (El Foro): Presentación e Instalación

Lo primero que haremos, como siempre, es visitar la página oficial del **phpbb** (www.phpbb.com) y pulsaremos sobre **downloads** (arriba en el centro), lo que nos conducirá a otra página desde la que descargaremos el archivo **phpbb 2.4 [Full Package] (versión ZIP)**



Quizás te sorprenda que solo hay un archivo a descargar y no hace distinciones entre Sistemas (Linux, Windows...). Eso es debido a que un programa creado en PHP será interpretado por un Servidor Web (esté instalado dicho Servidor sobre el Sistema que sea). Por lo tanto, un programa en PHP es independiente de la plataforma. Para que los puristas no se enfaden, diremos que esto no es 100% cierto, pero salvo que tengas un nivel de programación muy avanzado te aseguro que es no tendrás problemas en este sentido.

Una vez descargado lo descomprimimos donde queramos y obtendremos una carpeta llamada **phpBB2** y en su interior un montón de carpetas y archivos.



Procedamos a su instalación :)

1.- Inicia el APACHE y el Servidor MySQL. Recuerda que el Servidor MySQL se activa ejecutando el archivo c:\mysql\bin\winmysqladmin.exe

2.- Crea una carpeta en la raíz del Servidor Apache, en nuestro caso c:\apache\htdocs\, llámala por ejemplo **forophp**

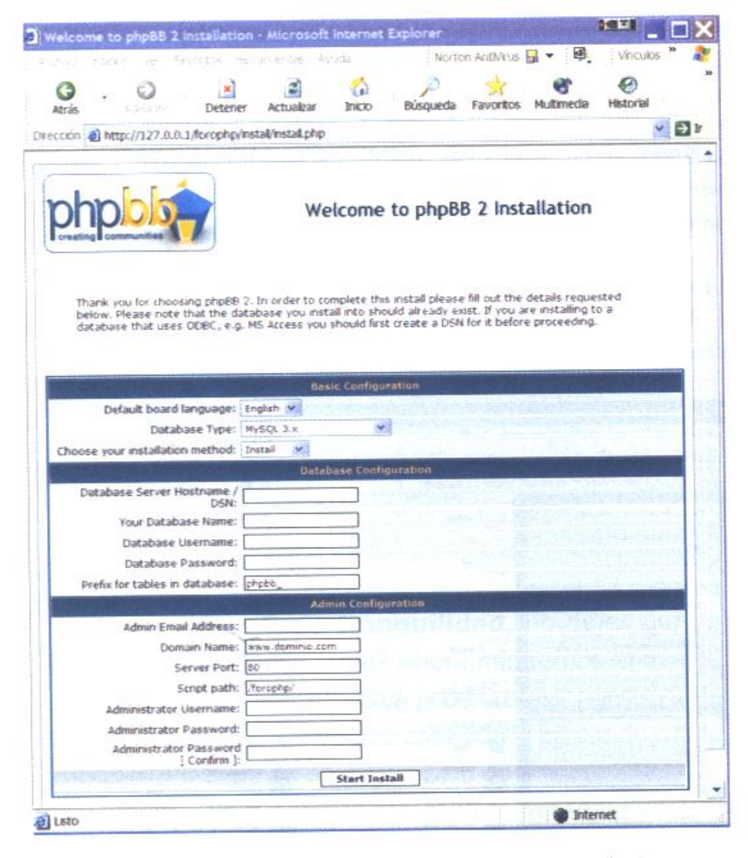
3.- Mete dentro de la carpeta c:\apache\htdocs\forophp\ TODO el contenido de la carpeta **phpBB2** (lo tienes en la imagen anterior)

4.- Ahora vamos a abrir nuestro navegador y accederemos por localhost (127.0.0.1) al archivo **C:\Apache\htdocs\forophp\install\install.php**

...venga hombre, no me mires así, que no es difícil. Simplemente abre el Internet Explorer (o tu navegador preferido) e introduce la página:

http://127.0.0.1/forophp/install/install.php (y cruza los dedos!!!)

Como ya debes saber (si has seguido los anteriores números), nuestro APACHE habrá atendido nuestra petición y gracias a que ahora puede leer páginas php, habrá interpretado y servido la página install.php ... ... ahora mismo delante de nosotros deberíamos tener la página de instalación del foro **phpbb**



A partir de este momento ya trabajaremos directamente con el navegador y el proceso de instalación del foro es bastante **trivial**, je, je... trivial es una palabra muy interesante que suele significar lo siguiente: Si no sabes hacer el resto te las arreglas por tu cuenta. Odio la palabra trivial, así que **nosotros no seremos triviales** y lo haremos paso a paso :)

- En language (idioma) seleccionaremos el único disponible: **Ingles** (después ya cambiaremos eso).
- En Database Type seleccionaremos la nuestra, recuerda que el Telnet te dio la versión: **MySQL 3.x**
- En Installation Method (método de instalación): **Install**

Ahora pasamos a la parte que configura el acceso a los datos, que por eso hemos instalado nuestro flamante Servidor MySQL

- Database Server Hostname / DSN sería el nombre de domio de la máquina donde estuviese el Servidor de Datos (o su IP) y el puerto para acceder. En nuestro caso pondremos nuestra IP y para que le funcione a todo el mundo pondremos la LOOP-IP 127.0.0.1 y el puerto 3306, es decir, escribiremos **127.0.0.1:3306**

- Database Name (Nombre de la Base de Datos): Podríamos poner cualquier nombre pero antes deberíamos crear una base en nuestro Servidor MySQL. Para no liarnos utilizaremos una base llamada mysql que MySQL crea durante su instalación, por lo tanto escribiremos **mysql**

- Database Username: No ponemos nada

- Database Password: No ponemos nada

- Prefix for... : Es el prefijo que antecede a cada tabla de datos que se creará en nuestro servidor MySQL. Imagina que instalas/creas 1000 tablas en la Base de Datos, sería un infierno saber a qué programa pertenece cada tabla, por eso es muy útil que tengan un prefijo (después lo veremos mejor). Nosotros vamos a dejarlo tal como está: **phpbb_**

Ahora pasamos a configurar los datos de Administración del Foro.

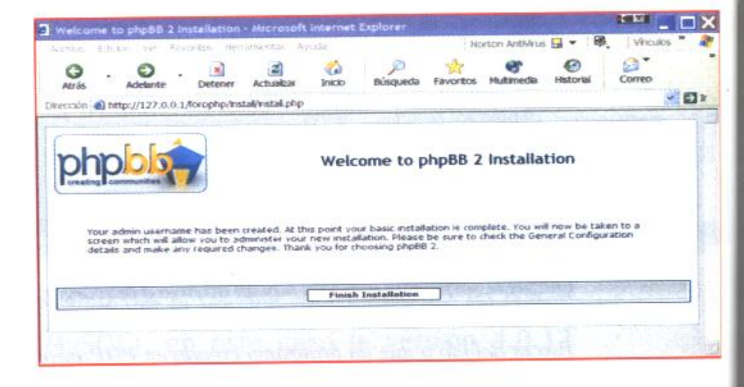
- Admin Email Address: Pon un mail real al que tengas acceso. El foro, bajo ciertas circunstancias, enviará un mail informativo al administrador (en este caso Tú). Nosotros ponemos prueba@menta.net

Domain Name: Esto es importante. Si quieres que tu foro esté disponible para Internet deberás poner tu IP Externa, si quieres que esté disponible para la Intranet de tu casa deberás poner tu IP Interna y si quieres únicamente que esté disponible para tu PC deberás poner la LOOP-IP (127.0.0.1)... podría darse otro caso, que tuvieses contratado un nombre de dominio asignado a tu IP externa, en ese caso deberías poner tu nombre de dominio para que todo el mundo tuviese acceso al foro. Nosotros utilizaremos la LOOP-IP, escribiremos **127.0.0.**

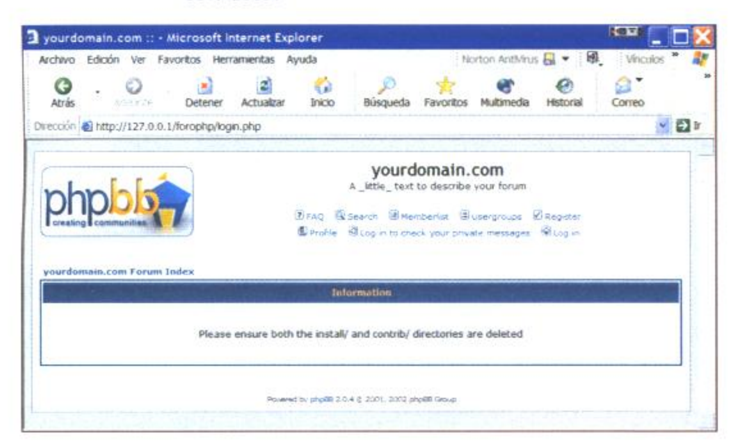
- Administrator Username: Es el Nombre de Usuario del Administrador del Foro, no te olvides de apuntarlo en un papel para que no se te olvide!!! En nuestro caso ponemos **attfe494**

- Administrator Password: Pues eso, el password del administrador, en nuestro caso **cedente1** (no te olvides de apuntartelo en algún sitio)

Finalmente pulsa **Start Install** y cruza los dedos para que todo funcione bien ;p, en cuyo caso saldrá la siguiente pantallita



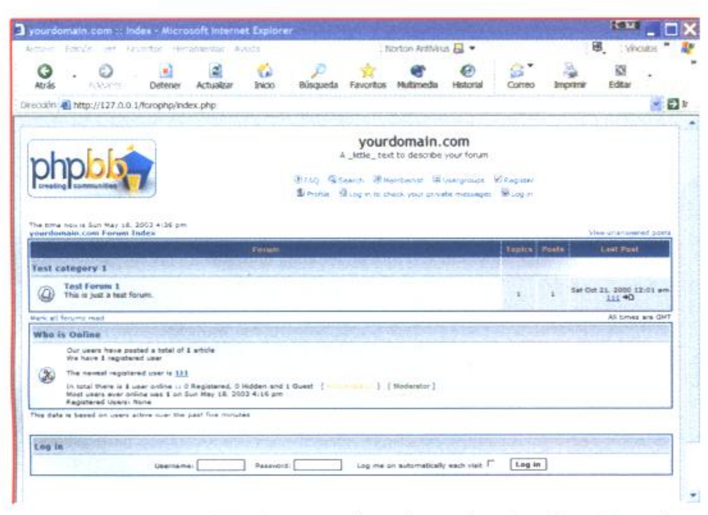
pulsaremos Finish Installation y nos encontraremos con esto:



Nos está diciendo que borremos las carpetas **contrib** y **install.** Si has seguido nuestros pasos, estas carpetas están en c:\apache\htdocs\forophp\... yo te recomiendo que no las borres, simplemente muévelas a una carpeta cualquiera de tu disco duro fuera del directorio c:\apache. La idea es que cuando quieras reinstalar o updatar el foro, puedas ponerlas de nuevo en el directorio c:\apache\htdocs\forophp\ y ejecutar la instalación desde el navegador (http://127.0.0.1/forophp/installx/install.php)

Ahora cierra la ventana del explorador y mueve las carpetas que te he dicho... hazlo!!! Si no lo haces no podrás entrar al foro ;p

Ahora vamos a acceder al foro. Abrimos el explorer e introducimos la dirección de acceso al foro: http://127.0.0.1/forophp/index.php y debería salirte la página inicial del foro:

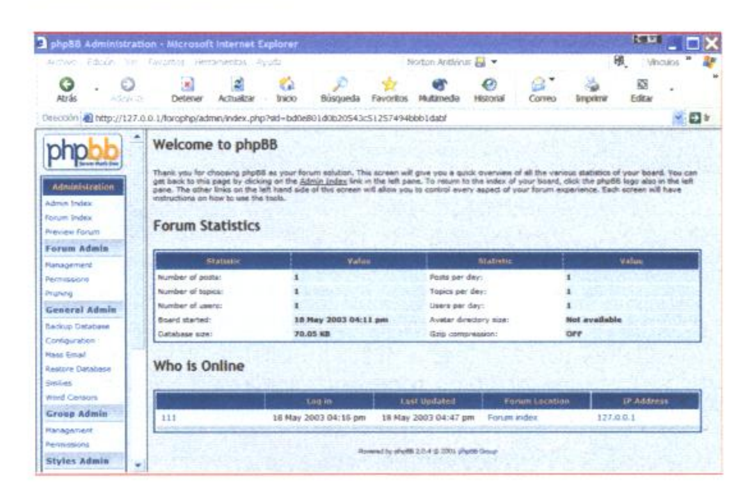


Arriba a la izquierda pulsa sobre **Log In** e introduce tu user y pass de Administrador, nosotros utilizamos **attfe494** y **cedente 1** respectivamente. Una vez introducidos ya estarás reconocido como ADMINISTRADOR DE TU PROPIO FORO ;)

Para administrar el foro, abajo encontrarás un enlace llamado **Go to Administration Panel**



Pues ya sabes, picando ese enlace llegarás a la Zona de Administración.



Finalizando... ...

A partir de ahora ya es cosa tuya, esto es como un lienzo en blanco en el que debes plasmar tu obra.
En la Web oficial puedes desde traducir el foro al idioma que desees hasta aplicarle un template (aspecto) de lo más variado de forma automática.

AGRADECIMIENTOS a los desarrolladores PHPBB

